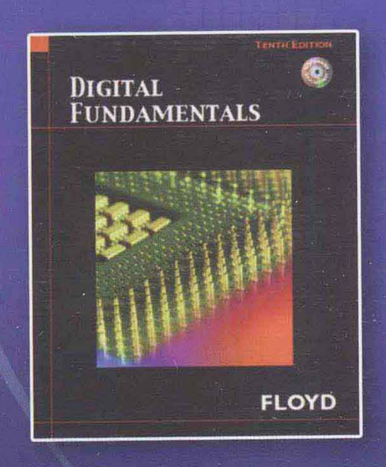
国外电子与通信教材系列

PEARSON

数字电子技术

（第十版）

Digital Fundamentals, Tenth Edition



[美] Thomas L. Floyd 著

余琢 译



http://www.phei.com.cn

数字电子技术（第十版）

Digital Fundamentals, Tenth Edition

本书是一本关于数字电子技术的经典教材，专门针对国内教学的实际情况进行了缩减。全书主要介绍了数字电子技术的基本概念、数字系统、逻辑门、布尔代数和逻辑化简、组合逻辑分析、组合逻辑电路函数、锁存器、计数器、移位寄存器、存储器、数字信号处理、集成电路技术等。全书的特色在于示例与习题丰富、图解清晰、语言流畅、写作风格简约。

配套网站及实验平台内容丰富，为学生理解所学知识提供了实践的机会。

本书可作为高等院校电子信息类专业本科生的教材，也可供相关技术、科研人员使用，或作为继续教育的参考用书。

译者简介

余璆　上海工程技术大学教师，长期讲授数字电路基础与电子学课程，具有丰富的教学经验。

本书配套网站：http://wps.prenhall.com/chet_floyd_digitalfun_10
http://www.hxedu.com.cn

本书教辅（PPT，习题解答，等等）的申请方式请参见书末的“教学支持说明”。



ISBN 978-7-121-13257-5
定价：59.80 元

ISBN 978-7-121-04396-3
定价：58.00 元



www.pearson.com





ISBN 978-7-121-19041-4
定价：65.00 元



策划编辑：冯小贝
责任编辑：冯小贝
责任美编：李　雯



国外电子与通信教材系列

数字电子技术

（第十版）

Digital Fundamentals, Tenth Edition

［美］ Thomas L. Floyd 著

余 璆 译

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书是一本关于数字电子技术的经典教材，并专门针对国内教学的实际情况进行了缩减。全书主要介绍了数字电子技术的基本概念、数字系统、逻辑门、布尔代数和逻辑化简、组合逻辑分析、组合逻辑电路函数、锁存器、计数器、移位寄存器、存储器、数字信号处理、集成电路技术等。全书的特色在于示例与习题丰富、图解清晰、语言流畅、写作风格简约。

本书可作为高等院校电子信息类相关专业本科生的教材，也可供相关技术、科研人员使用，或作为继续教育的参考用书。

版权贸易合同登记号　图字：01-2012-1496

图书在版编目(CIP)数据

数字电子技术：第 10 版/(美)弗洛伊德(Floyd,T. L.)著；余璆译. —北京：电子工业出版社，2014.1

(国外电子与通信教材系列)

书名原文：Digital Fundamentals

ISBN 978-7-121-19041-4

Ⅰ. ①数…　Ⅱ. ①弗… ②余…　Ⅲ. ①数字电路-电子技术-高等学校-教材　Ⅳ. ①TN79

中国版本图书馆 CIP 数据核字(2012)第 281888 号

策划编辑：冯小贝

责任编辑：冯小贝

印　　刷：涿州市京南印刷厂

装　　订：涿州市京南印刷厂

出版发行：电子工业出版社

北京市海淀区万寿路 173 信箱　邮编　100036

开　　本：787×1092　1/16　印张：32.5　字数：917 千字

印　　次：2014 年 1 月第 1 次印刷

定　　价：65.00 元

凡所购买电子工业出版社的图书有缺损问题，请向购买书店调换；若书店售缺，请与本社发行部联系。联系及邮购电话：(010)88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：(010)88258888。

译　者　序

数字电子技术是工科电类专业的一门专业基础课，尤其是对自动化和计算机专业而言，这门课对后续课程的进一步学习非常重要。又由于计算机、集成电路的不断发展和数字电路本身层出不穷的应用，使得这门学科不断地得到新的充实，因此有必要翻译引进这方面的国外优秀教材。

本书译自电子工业出版社出版的《数字电子技术（第十版）（英文版）》一书。该英文版改编自 Thomas L. Floyd 所著的 *Digital Fundamentals*，*Tenth Edition*。原著的内容和结构与国内的数字电子技术教材比较吻合，在这一基础上考虑到内容的精简，以使得它更符合国内教学的要求和课时量，因此推出了英文改编版，作为一本双语教材供学校选用。

这次又在英文改编版的基础上进行了翻译，其一是希望把国外的数字电子技术教材介绍进来，因为它的教学内容和国内的教材既有相同之处，也有它的丰富特性。比如，原理的讲解较为详细和清晰，一些章节带有实例或综合举例；同时结合数字逻辑电路，介绍一些计算机小知识。第二个目的是此书可以作为双语教材《数字电子技术（第十版）（英文版）》的参考书，希望能够帮助教师和学生完成双语教学的教与学的任务。当然，此书也可以作为应用数字电子技术的相关人员的参考书。

由于译者的水平有限，一定会有一些错误和不妥之处，恳请读者不吝指教。

目　录

第1章　基本概念

章节提纲

1.1　数字量与模拟量

1.2　二进制数、逻辑电平和数字波形

1.3　固定功能的集成电路

访问本书配套的网址

在网址 http://www.prenhall.com/floyd 中可以得到本书的辅导资料①。

1.1　数字量与模拟量

模拟量具有连续的数值，数字量具有离散的数值。自然界中大多数可以测量的事物都以模拟量的形式出现。例如，空气温度在一个连续的范围内变化。在给定的一天里，温度不会立即从70℉上升到71℉；这中间经历无数个温度值。如果绘制一个典型的夏季温度图，将会得到一个平滑和连续的曲线(类似于图1.1的曲线)。其他模拟量的例子是时间、压力、距离和声音。

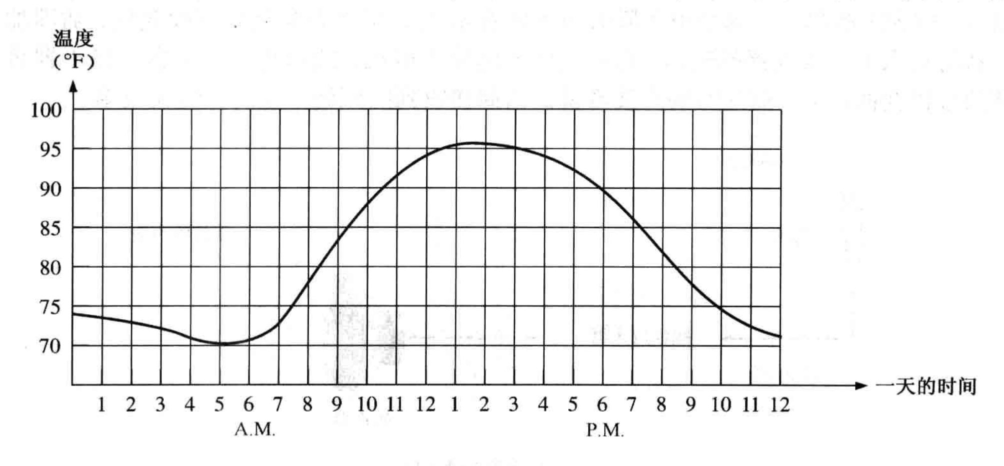


图1.1　模拟量图(温度/时间)

相对于一个连续的温度图，假设每小时测量一次温度。现在有一个24小时内每隔一小时采样测量到的离散温度值，如图1.2所示，这样就可以有效地将模拟量转换成数字量的形式，即用一个个数字码对应于每个采样到的温度值。注意，图1.2本身并不是模拟量的数字表示。

数字量的优点　在电学应用方面，数字量表示法和模拟量表示法相比有一定的优势。其一，数字数据和模拟数据相比，前者在处理和传输方面更有效、更可靠。其二，数字数据在需要保存时，更显示了它的优越性。例如，转换成数字形式的音乐，要比相对应的模拟形式更简洁，复制时更精确、更清晰。噪声(不需要的电压波动)几乎不会影响数字数据，但会影响模拟信号。

① 相关资料也可登录 www.hxedu.com.cn 获取，本书教辅(PPT，习题解答)的申请方式请参见书末的"教学支持说明"。

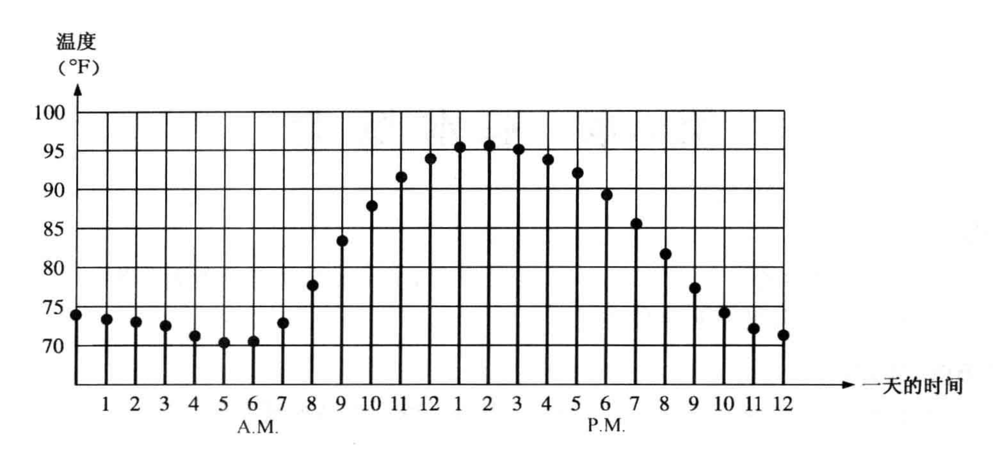


图 1.2　图 1.1 的模拟量样本值的表示法(量化)。每个值由点表示,它可以由一些0和1组成的数字码表示

1.1.1　模拟电子系统

扩音系统用于把声音放大,从而让更多的听众听到,这是模拟电子应用的一个简单例子。图 1.3 的基本图示给出了自然界中的模拟量,即声波,它由麦克风接收,并将其转换为较弱的模拟电压,称为声频信号。这个电压随着声波的音量大小和频率变化而连续变化,随即加到线性放大器的输入中。放大器的输出,也就是放大的输入电压,随即传入扬声器。扬声器将放大的音频信号再变回声波,而这时的声波音量比话筒接收到的原始声波的音量大很多。

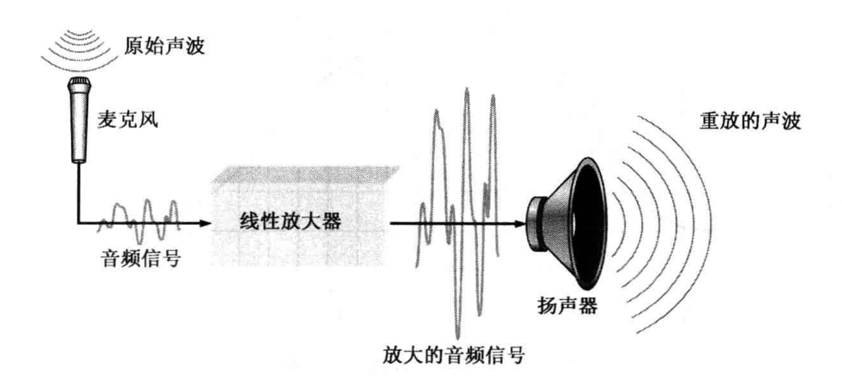


图 1.3　基本的声音广播系统

1.1.2　使用数字方法与模拟方法的系统

光盘(CD)播放器是一个同时使用数字电路和模拟电路的系统。图 1.4 的简单框图给出了它的基本原理。数字格式的音乐存储在光盘上。激光二极管光学系统接收旋转光盘上的数字数据,然后传送到数-模转换器(DAC)中。数-模转换器将这些数字数据转换成模拟信号,即原来音乐电子意义上的再现。线性放大器把模拟信号放大并传送到扬声器,以供欣赏。在将音乐存储在一张 CD 上时,过程基本上和以上描述的相反,这时使用模-数转换器(ADC)。

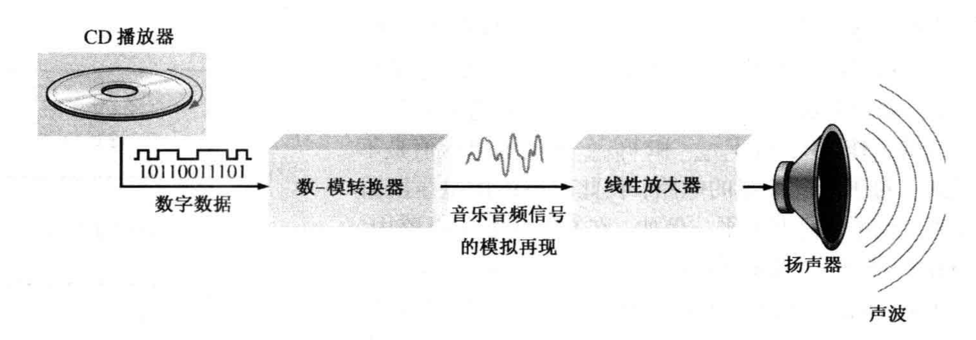


图1.4 CD播放器的基本框图，这里只画出了一个通道

1.1节 温故而知新 （答案在本章的结尾。）

1. 定义模拟量。
2. 定义数字量。
3. 解释数字量和模拟量的不同之处。
4. 给出一个使用模拟量的系统的例子，以及给出一个同时使用数字量和模拟量的系统的例子。给出一个完全使用数字量的系统。

1.2 二进制数、逻辑电平和数字波形

1.2.1 二进制数

二进制系统中的两个数——1和0，称为位(比特，bit)，是二进制数(binary digit)的缩写。在数字电路中，使用两个不同的电压电平表示这两个位。一般情况下，高电压用1来表示，低电压用0来表示。这称为正逻辑，本书将都使用正逻辑。

高电压(H)=1　　低电压(L)=0

在另一种系统中，1表示低电压，0表示高电压，这称为负逻辑。

一组位(一些1和0的组合)称为码，用来表示数字、字母、符号、指令及任何给定应用中的对象。


计算机小知识

数字计算机的概念可以追溯到Charles Babbage，他在19世纪30年代发明了一台原始的机械式计算器。John Atanasoff于1939年首先在数字计算机中使用了电子处理方法。1946年，第一台使用真空电子管电路并称为ENIAC的数字计算机诞生。尽管它的体积占据了整个房间，但是ENIAC的计算能力还不如我们现在常用的计算器。

1.2.2 逻辑电平

用来表示1和0的电压称为逻辑电平。理想情况下，一个电平表示高电压，另一个电平表示低电压。在实际的数字电路中，这个高电压可以是指定的最小值和最大值之间的任意值。同样，低电压也可以是指定的最小值和最大值之间的任意值。在指定的高电平范围和低电平范围之间是不能有重叠的。

图 1.5 给出了数字电路中高电平和低电平的通常范围。变量 $V_{H(max)}$ 表示高电平的最大值，变量 $V_{H(min)}$ 表示高电平的最小值。$V_{L(max)}$ 表示低电平的最大值，$V_{L(min)}$ 表示低电平的最小值。在正常的工作情况下，$V_{L(max)}$ 和 $V_{H(min)}$ 之间的电压值是不可以出现的。对于一个给定的电路，在此范围内的电压既可以是高电平，也可以是低电平。例如，在 CMOS 数字电路中，高电平值在 2 ~ 3.3 V，低电平压值在 0 ~ 0.8 V，也就是说，如果使用 2.5 V，电路将把它看成是高电平或二进制 1。如果使用 0.5 V，那么就是低电平或二进制 0。对于这种类型的电路，0.8 ~ 2 V 的电平值是不可以出现的。

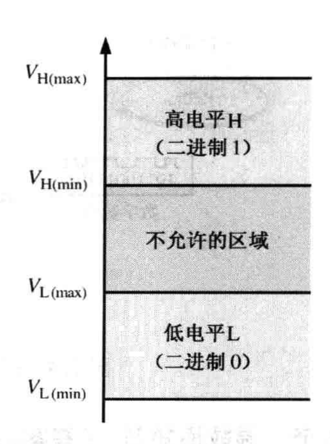


图 1.5　数字电路逻辑电平的电压范围

1.2.3　数字波形

数字波形由两种不同的电平值组合而成，它们在高、低电平或状态之间不断地变化。图 1.6(a) 给出一个正向脉冲，是在电压(或电流)从低电平变到高电平，再从高电平变回到低电平时产生的。图 1.6(b) 给出一个反向脉冲，是在电压从高电平变到低电平，再从低电平变回到高电平时产生的。数字波形由这一系列的脉冲组成。

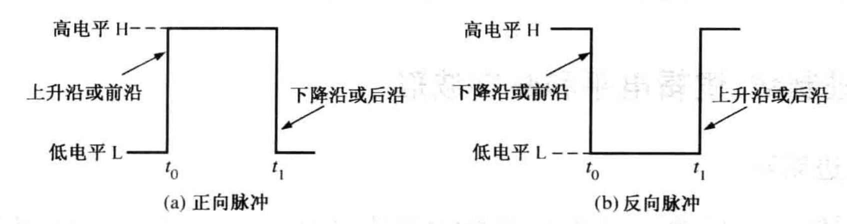


图 1.6　理想的脉冲

脉冲　如图 1.6 所示，脉冲有两个边沿：在 t_0 时刻首先出现的为前沿，在 t_1 时刻随后出现的为后沿。对于一个正向脉冲，前沿是上升沿，后沿是下降沿。图 1.6 所示的脉冲是理想状态下的脉冲，因为假设上升沿和下降沿的变化是没有时间范围的(瞬间)。实际情况是，这些变化是有时间范围的，尽管大多数的数字波形可以假定为理想脉冲。

图 1.7 给出了一个非理想的脉冲。实际上，所有脉冲或多或少都存在这些非理想的特性。通常，杂散电感和电容效应会产生超调量和振荡。杂散电容和电路电阻会产生下调量，形成时间常数不大的 RC 电路。

从低电平到高电平所需的时间称为上升时间 t_r，从高电平到低电平所需的时间称为下降时间 t_f。在实际运用中，通常测量的上升时间是从脉冲幅度(相对于基线的高度)的 10% 处到脉冲幅度的 90% 处的时间宽度，测量的下降时间则是从幅度的 90% 处到幅度的 10% 处的时间宽度。如图 1.7 所示，上升时间和下降时间不包括脉冲顶部和底部的 10%，因为这部分区域的波形是非线性的。脉冲的宽度 t_W 就是脉冲的持续时间，通常把上升沿和下降沿幅度 50% 处的时间间隔定义为脉冲宽度，如图 1.7 所示。

波形特性　在数字系统里，遇到的大多数波形都是由一系列的脉冲组成的，有时称为脉冲序列，它们可以分为周期的和非周期的。周期波形就是在一个固定的时间间隔里不断重复自身，这个时间间隔称为周期(T)。频率(f)是重复的速率，测量单位是赫兹(Hz)。而一个非周期性脉冲波形则不会在一个固定的时间间隔里重复，它可能由脉冲宽度不确定的脉冲组成，也有可能由时间间隔不确定的脉冲组成，图 1.8 给出两种波形的例子。

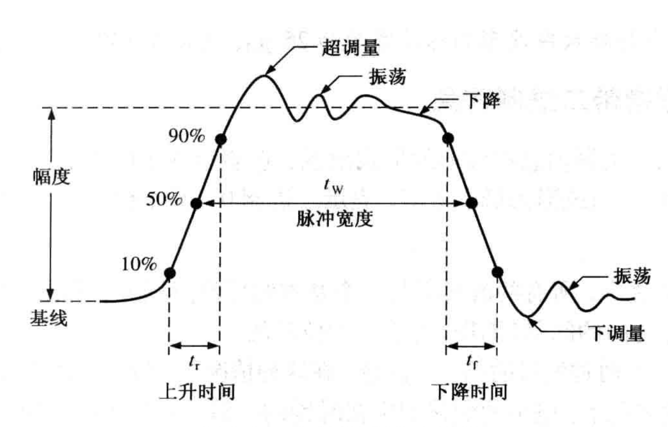


图 1.7　非理想脉冲的特性

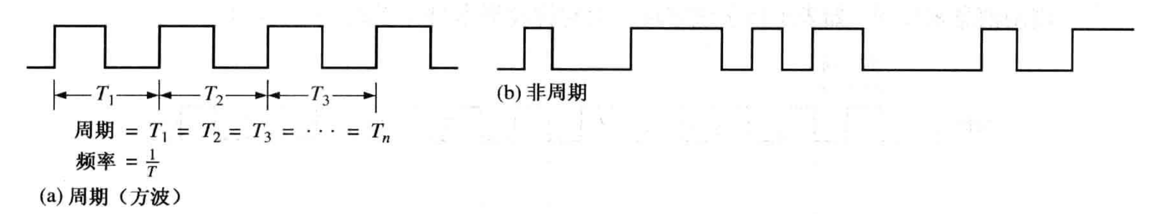


图 1.8　数字波形的例子

脉冲(数字)波形的频率(f)就是其周期(T)的倒数，它们之间的关系如下所示：

$$f = \frac{1}{T} \tag{1.1}$$

$$T = \frac{1}{f} \tag{1.2}$$

周期数字波形的一个重要特性就是它的占空比，它是脉冲宽度(t_W)和周期(T)的比值，可以用百分比来表示

$$\text{占空比} = \left(\frac{t_W}{T}\right) \times 100\% \tag{1.3}$$

例 1.1　图 1.9 为一个周期数字波形的一部分，单位为毫秒(ms)，试计算：(a)周期；(b)频率；(c)占空比。

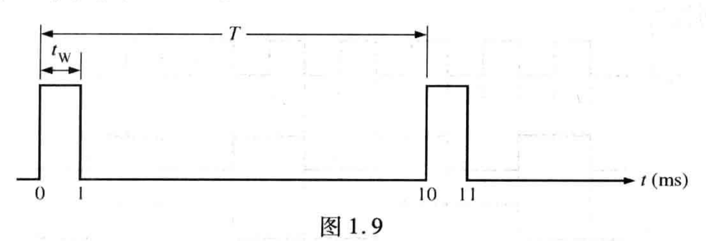


图 1.9

解：(a)周期是从一个脉冲沿到下一个相对应的脉冲沿的时间。这里周期 T 就是上升沿到上升沿的时间，如图 1.9 所示，T 等于 **10 ms**。

(b) $f = \frac{1}{T} = \frac{1}{10\ \text{ms}} = \mathbf{100\ Hz}$

(c) 占空比 $= \left(\frac{t_W}{T}\right) \times 100\% = \left(\frac{1\ \text{ms}}{10\ \text{ms}}\right) \times 100\% = \mathbf{10\%}$

相关问题[①]：一个周期数字波形的脉冲宽度为25 μs，周期为150 μs，试求出频率和占空比。

1.2.4 数字波形携带二进制信息

数字系统处理的二进制信息以波形的形式出现，它表示顺序序列的二进制位。当波形为高电平时，表示二进制1；当波形为低电平时，表示二进制0。每个位在一个序列里所占的固定时间间隔称为位时间。

时钟 在数字系统中，所有的波形都与一个基本时序波形同步，称之为时钟(clock)。时钟是周期波，每个脉冲之间的间隔(周期)等于一个位时间。

图1.10所示为一个时钟波形的例子。注意，在这种情况下，波形*A*的电平变化都发生在时钟波形的前沿。在其他情况下，电平的变化发生在时钟的后沿。在每个位时间之内，波形*A*可为高电平也可为低电平。这些高电平和低电平组成了图1.10所示的位序列。若干位组成一组就可作为一个二进制信息来使用，如表示数字或字母。而时钟波形本身并不携带任何信息。

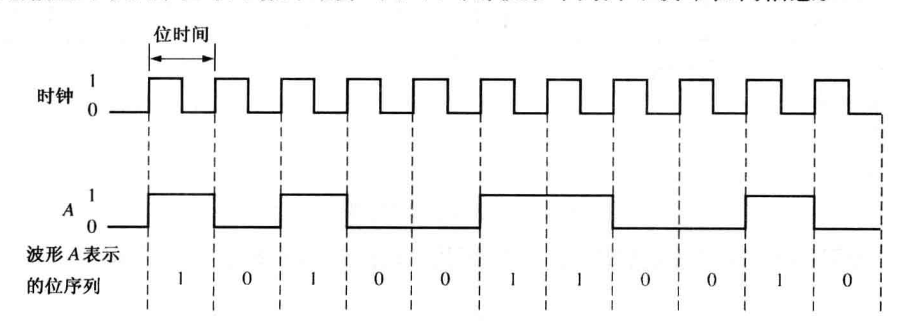


图1.10 时钟波形和位序列表示的波形同步的例子



计算机小知识

计算机的运行速度取决于系统使用的微处理器的类型。计算机运行速度的定义就是微处理器工作的时钟频率的最大值，例如3.5 GHz。

时序图 时序图就是数字波形的图形，它表示两个或两个以上波形的实际时间关系，还表示波形和波形之间的相互变化关系。图1.11给出了4个波形组成的时序图的例子。从这个时序图可以确定相互关系。例如，波形*A*、*B*和*C*仅在位时间7时同为高电平(阴影部分)，在位时间7结束时变回到低电平。

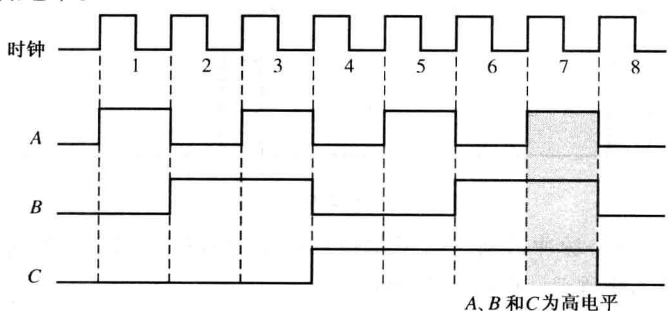


图1.11 时序图的例子

① 答案在本章的结尾。

1.2.5　数据传送

数据是指一组可以用来传递某种信息的位。使用数字波形表示的二进制数据，必须在数字系统中从一个电路传送到另一个电路，或者从一个系统传送到另一个系统，以实现某个设定的目的。例如，计算机存储器中的数字是以二进制的形式存储的，它必须传送到计算机的中央处理器才能实现加法运算。然后加法运算的结果必须传送到显示器显示并且/或者回送到存储器中。如图1.12所示，二进制数据的传送方式有两种——串行和并行。

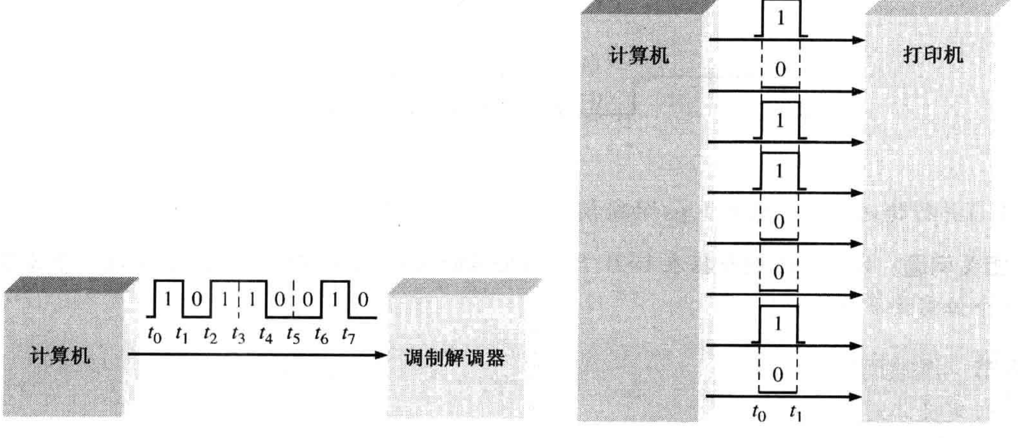


图1.12　二进制数据的串行和并行传送的图例。这里只显示数据线

图1.2(a)为计算机传送数据到调制解调器的例子，这时位以串行的方式从一个点传送到另一个点，沿着一条导线每次传送一位。在 t_0 到 t_1 这段时间间隔里，送出第一位。在 t_1 到 t_2 这段时间间隔里，送出第二位，以此类推。若要串行输出8位，则需花费8个时间间隔。

当若干位以并行的方式传送时，一组位中的每一个位可以同时通过不同的线路传送。图1.12(b)为8位数据从计算机传送到打印机的例子。和串行传送需要8个时间间隔相比，并行方式传送8个位只需要一个时间间隔。

综上所述，二进制数据串行传送的优点是所需要的数据线最少，即只需要一条线路。而在并行传送中，线路的数量等于一次传送的位的数量。串行传送的缺点是相对于并行传送，它需要更长的时间来完成一个给定位数的传送。例如，如果1 μs可以传送一位，那么需8 μs来完成串行传送8个位，但并行传送8个位只需1 μs。并行传送的缺点是和串行相比，需要更多的线路。

例1.2　(a)试给出图1.13波形 *A* 中串行传送8个位所需要的时间，并指出位的顺序。最左边的位先开始传送。以1 MHz的时钟频率作为基准。

(b)在并行传送中传送同样的8个位需要多少时间？

解：(a)因为时钟的频率为1 MHz时，周期为

$$T = \frac{1}{f} = \frac{1}{1\ \text{MHz}} = 1\ \mu\text{s}$$

传送每位需要1 μs，传输8个位则需要

$$8 \times 1\ \mu\text{s} = \mathbf{8\ \mu s}$$

为了确定每个位的顺序，必须确定图1.13的每个位时间内的波形。如果波形A在位时间内为高电平，则传送1。如果波形A在位时间内为低电平，则传送0。位的顺序如图1.14所示。从最左边的位最先开始传送。

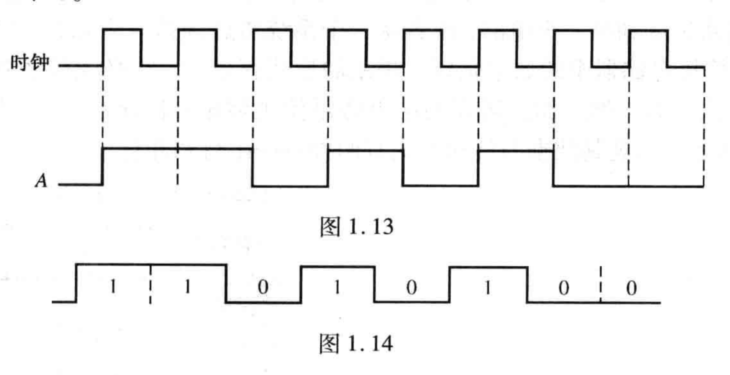


图1.13

1 1 0 1 0 1 0 0

图1.14

(b)并行传送8个位需要**1 μs**的时间。

相关问题：如果二进制数据在USB上以每秒480兆位(480 Mbps)的速度传送，那么串行传送16个位需要多少时间？

1.2节　温故而知新

1. 定义二进制。
2. 位的概念是什么？
3. 在二进制系统中位的组合是什么？
4. 一个脉冲的上升和下降时间是如何计算的？
5. 如果确定一个波的周期，那么频率是如何计算的。
6. 解释时钟波形的概念？
7. 时序图的作用是什么？
8. 二进制数据的并行传送与串行传送相比的优点是什么？

1.3　固定功能的集成电路

单片集成电路(IC)是一个完全由单个小型的硅芯片组成的电子电路。组成电路的所有元件——晶体管、二极管、电阻和电容是单个芯片的一部分。固定功能的逻辑和可编程的逻辑是数字集成电路的两大类型。在固定逻辑功能的芯片中，逻辑功能已由开发商写入，不能改变。

图1.15给出了一个固定功能集成芯片封装类型的剖面图，其中露出了封装内部的芯片电路，标出了芯片连接封装引脚用以连接外部的各种电路。

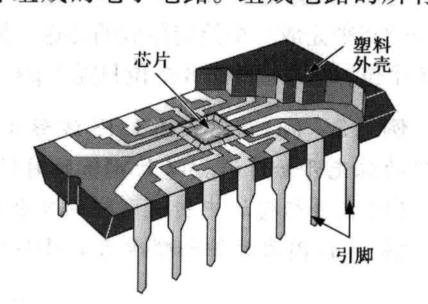


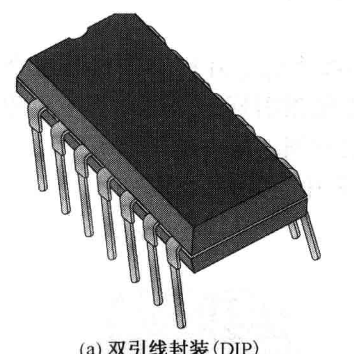
图1.15　一个固定功能集成芯片封装类型的剖面图(双引线封装)，给出了芯片的内部连接输入/输出引脚的封装部分

1.3.1　集成电路封装

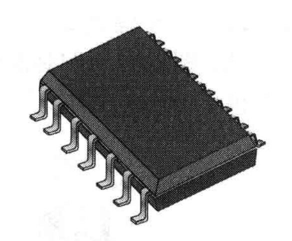
集成电路封装以它们安装在印制电路板(PCB)上的方法来分类，例如对穿孔封装和表面贴装等类型。对

穿孔封装类型的引脚通过穿孔插入印制电路板，以便可以和另一边的导体焊接。最为普通的对穿孔封装是双引线封装(DIP)，如图1.16(a)所示。

另一种集成电路封装的类型是使用表面贴装技术(SMT)。表面贴装相对于对穿孔封装节省空间。通过印制电路板的孔对于表面贴装技术是多余的。表面贴装的引脚直接焊接在印制电路板一面的导体上，留出另一面用于附加的电路。其次，对于有相同引脚数的电路，表面贴装与双引线封装相比尺寸更小，因为表面贴装的引脚分布更紧凑。一个表面贴装的例子就是小轮廓集成电路(SOIC)，如图1.16(b)所示。



(a) 双引线封装(DIP)



(b) 小轮廓集成电路(SOIC)

图1.16 对穿孔和表面贴装芯片的例子。相同引脚数的DIP比SOIC的尺寸大，DIP大约为0.785英寸①长，SOIC大约为0.385英寸长

在一种尺寸范围内可用的表面贴装技术的不同类型，取决于引脚数(越复杂的电路和引线配置需要更多的引脚)。图1.17给出了几种类型的例子。如图所见，收缩小轮廓封装(SSOP)的引脚形成鸥翼式形状。带引线的塑料芯片载体(PLCC)的引脚在封装下以J形翻转。无引线陶瓷芯片(LCC)具有金属接触面，焊接在它的陶瓷体上。薄型四侧引脚扁平封装(LQFP)也具有鸥翼式形状的引脚。片状刻度封装(CSP)和精密间距球形网格阵列(FBGA)的引脚嵌入在封装的底部相互接触。

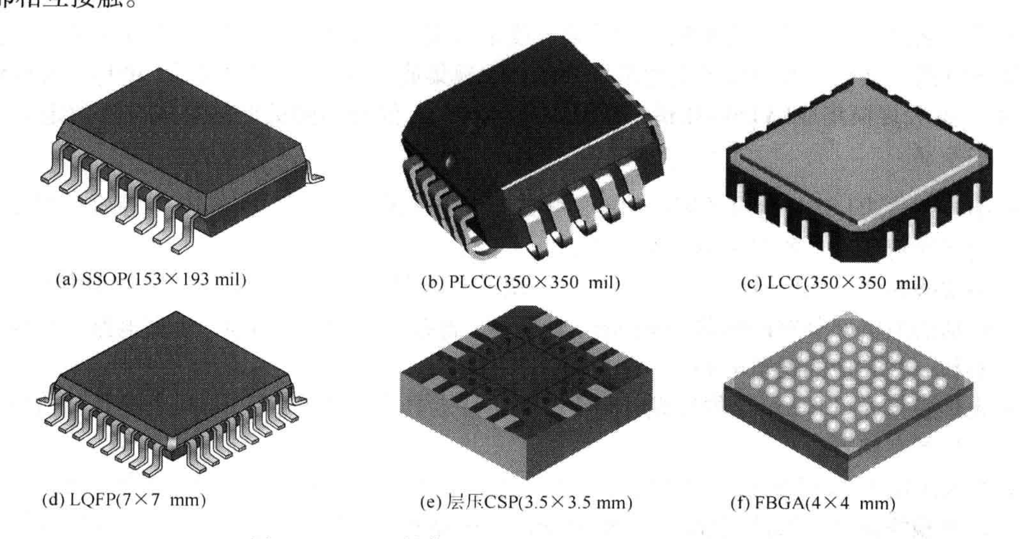


图1.17 SMT封装配置，(e)和(f)给出的是底视图

① 1英寸=2.54厘米。

1.3.2 引脚编号

所有集成电路的封装都具有一个标准形式的引脚编号。双引线封装(DIP)和收缩小轮廓封装(SSOP)的引脚都具有一定的编号排列方式，如图 1.18(a)的 16 个引脚封装的芯片所示。观察芯片封装的顶部，引脚 1 有一个标识，可以是一个小圆点、一个缺口或是一个斜角边。小圆点总是紧靠着引脚 1。同样，缺口方向朝上，引脚 1 总是在左上角的位置，如图所示。从引脚 1 开始，逐步向下引脚编号递增，然后跨越芯片在另一边编号向上递增。最大的引脚编号总是在缺口的右边小圆点对面的位置。

带引线的塑料芯片载体(PLCC)封装和无引线陶瓷芯片(LCC)封装在全部的 4 个边上都具有引脚。引脚 1 由一个小圆点标记，或由其他的索引标记，它处于一边的中心位置。引脚的编号沿逆时针递增，图中看到的是封装顶部的视图。最大的引脚编号总是处于引脚 1 的右边。图 1.18(b)给出了 20 个引脚的 PLCC 封装芯片的引脚分布。

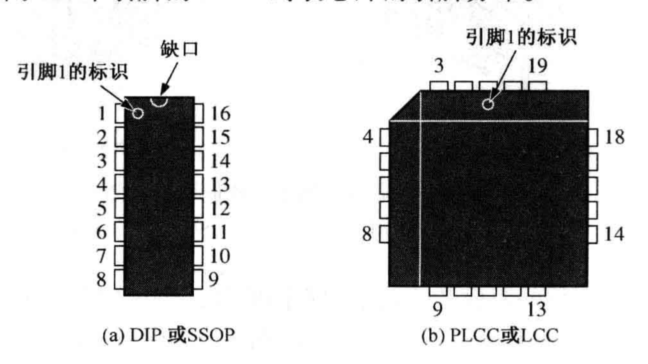


图 1.18 集成电路封装引脚的标准类型。给出的是顶视图

1.3.3 固定功能集成电路的集成度的分类

固定功能的数字集成电路按照它们的集成度来分类。这里列出了从集成度最低到集成度最高的集成电路。其中涉及的集成度通常划分为小规模集成(SSI)、中规模集成(MSI)、大规模集成(LSI)、甚大规模集成(VLSI)和超大规模集成(ULSI)，但是不同类型的集成芯片的定义可能会有一些差别。

- **小规模集成(SSI)**给出的固定功能的集成电路在一块芯片上最多有 10 个等效门电路，它们包括若干基本的门电路和触发器。
- **中规模集成(MSI)**给出的集成电路在一块芯片上具有 10 个到 100 个等效门电路。其中包括的逻辑部分有译码器、编码器、计数器、寄存器、多路复用器、运算电路、小型存储器和一些其他的逻辑电路。
- **大规模集成(LSI)**给出的集成电路在一块芯片上具有 100 个到 10 000 个等效门电路，其中包括存储器。
- **甚大规模集成(VLSI)**给出的集成电路在一块芯片上具有 10 000 个到 100 000 个等效门电路。
- **超大规模集成(ULSI)**电路包括容量很大的存储器、大型的微处理器和大型的单片计算机。在一块芯片上的集成度多于 100 000 个等效门的电路定义为超大规模集成电路。

1.3 节　温故而知新

1. 什么是集成电路?
2. 定义术语:DIP,SMT,SOIC,SSI,MSI,LSI,VLSI,ULSI。
3. 通常,固定功能的集成电路按照以下的等效门数量应该归为哪种类型?
(a)10 (b)75 (c) 500 (d)15 000 (e) 200 000

关键词

模拟量　连续的量或具有连续的值。

二进制数　有两个值或状态，描述为基为 2 和使用 1 和 0 作为其数字量的数字系统。

位(比特)　一个二进制数，可以是 1 或 0。

时钟　数字系统中的基本时序信号，用于同步运行的周期波。

编译程序　控制设计流程的程序，它把源代码翻译成目标代码，即一种可以逻辑测试或可以下载到目标装置的代码格式。

数据　以数字、字母或其他形式组成的信息。

数字量　和数字或离散量有关，具有不连续的值。

占空比　数字波形的脉冲宽度和周期的比，用百分比来表示。

固定功能逻辑　数字集成电路的一类，其中电路的功能不可以改变。

门　一种进行基本逻辑运算的逻辑电路，例如与门或者或门。

输入　进入一个电路的信号或引线。

集成电路(IC)　一种把所有的器件集成在一块非常小的半导体材料的芯片上的电路。

输出　离开电路的信号或引线。

并行　在数字系统中，数据在几条线路上同时出现，数据位同时得到传送或处理。

可编程逻辑　数字集成电路的一类，具有进行编程的特定功能。

脉冲　在一个称为脉冲宽度的时间里从一个电平到另一个电平的突然变化，紧接着又一个突然变化，变回到原来的电平。

串行　一个接着一个，例如串行传送数据位，即顺序传送而不是同时传送。

时序图　一些数字波形图，给出两个或多个数字波形之间的时序关系。

判断题 (答案在本章的结尾。)

1. 一个模拟量具有一个连续的值。
2. 一个数字量具有 10 个离散的值。
3. 在二进制系统中有两个数字。
4. 术语位(比特)是二进制数的缩写。
5. 在正逻辑中，低电平表示二进制 1。
6. 如果一个脉冲波的周期增加，那么它的频率也增加。
7. 一个时序图给出两个或多个数字波形之间的时间关系。
8. 如果一个反相器的输入为 1，那么它的输出为 0。

自测题 (答案在本章的结尾。)

1. 有连续值的量是
   (a)数字量 (b)模拟量 (c)二进制量 (d)自然量
2. 位的含义是
   (a)少量的数据 (b)一个1或一个0 (c)二进制数 (d)答案(b)和(c)
3. 脉冲前沿位于幅度10%和90%之间的时间间隔是
   (a)上升时间 (b)下降时间 (c)脉冲宽度 (d)周期
4. 在一个给定的波形中,每隔10 ms出现一个脉冲。则频率为
   (a)1 Hz (b)1 Hz (c)100 Hz (d)10 Hz
5. 在一个给定的数字波形中,其周期为脉冲宽度的两倍,则占空比为
   (a)100% (b)200% (c)50%
6. 一个LSI芯片具有的电路的集成度从
   (a)10到100个等效门 (b)100到10 000个等效门
   (c)2000到5000个等效门 (d)10 000到100 000个等效门

习题 (奇数题的答案在本书的结尾。)

1.1节 数字量与模拟量

1. 数字数据与模拟数据相比的优点是什么?给出两个。
2. 除了温度与声音之外,请给出一个模拟量。
3. 列出三个常用产品,它们既有数字也有模拟输出。

1.2节 二进制数、逻辑电平和数字波形

4. 解释正逻辑和负逻辑的区别。
5. 确定下列每个电平的序列,用位(1和0)表示:
   (a)高、高、低、高、低、低、低、高
   (b)低、低、低、高、低、高、低、高、低
6. 列出下列以位的序列表示的电平(高和低)的序列:
   (a)1011101 (b)11101001
7. 如图1.19给出的脉冲,在图上标出
   (a)上升时间 (b)下降时间
   (c)脉冲宽度 (d)幅度
8. 如图1.20所示,确定数字波形的周期。
9. 如图1.20所示,求波形的频率。
10. 图1.20中的脉冲波形是周期性的还是非周期性的?
11. 如图1.20所示,求波形的占空比。
12. 如图1.21所示,设位时间为1 μs,确定波形代表的位序列。
13. 如图1.21所示,求串行传送8个位所花费的时间及并行传输8个位所花费的时间。

图1.19

1.3节 固定功能的集成电路

14. 固定功能数字集成电路芯片的集成度有200个等效门电路,如何定义它?

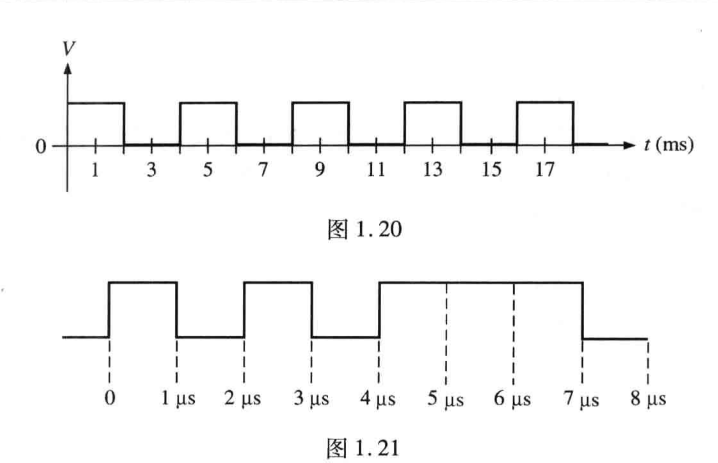


图 1.20

图 1.21

15. 解释 DIP 和 SMT 封装的主要区别。

16. 标出图 1.22 中的芯片封装的引脚编号，图中给出的是顶视图。

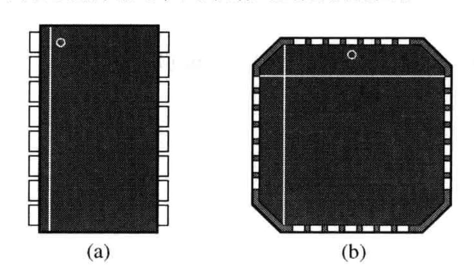


图 1.22

答案

温故而知新

1.1 节　数字量与模拟量

1. 模拟量的意思是连续的值。
2. 数字量的意思是离散的值。
3. 一个数字量有一个离散的值，一个模拟量有一个连续的值。
4. 一个公共广播系统是模拟系统。一个 CD 播放器是具有模拟量和数字量的系统。一台计算机只处理数字量。

1.2 节　二进制数、逻辑电平和数字波形

1. 二进制的意思是具有两种状态或两种值。
2. 一个位就是二进制数。
3. 就是 1 和 0 的组合。
4. 上升时间从信号幅度的 10% 到 90%，下降时间从信号幅度的 90% 到 10%。
5. 频率是周期的倒数。
6. 一个时钟波形是一个基本的时序波形，其他的波形由它导出。
7. 一个时序图给出两个或多个波形之间的时序关系。
8. 并行传输比串行传输快。

1.3 节 固定功能的集成电路

1. IC是所有元件集成在单个硅片上的电子电路。
2. DIP—双引线封装；SMT—表面贴装技术；SOIC—小轮廓集成电路；SSI—小规模集成；MSI—中规模集成；LSI—大规模集成；VLSI—甚大规模集成；ULSI—超大规模集成。
3. (a) SSI (b) MSI (c) LSI (d) VLSI (e) ULSI

例题的相关问题

1.1 $f = 6.67$ kHz；占空比 = 16.7%。

1.2 串行传输：3.33 ns。

判断题

1. T 2. F 3. T 4. T 5. F 6. F 7. T 8. T

自测题

1. (b) 2. (d) 3. (a) 4. (c) 5. (c) 6. (b)

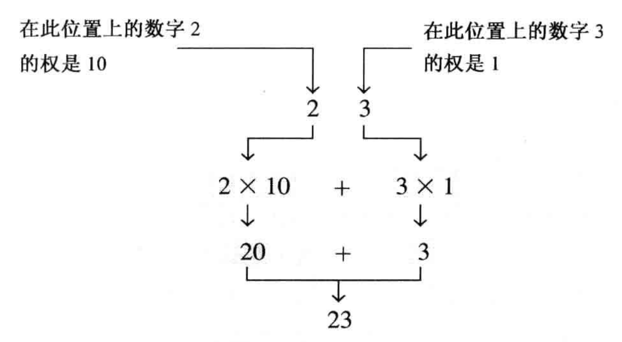
第 2 章　数字系统、运算和编码

章节提纲

2.1　十进制数

◇ 十进制数字系统有 10 个数字。

在十进制数字系统中，10 个数字中的每一个数字(从 0 到 9)都表示某个数量。这 10 个符号(数字)并没有限制仅仅表示 10 个不同的数字量，因为可以在数字量相应的位置上分别使用这 10 个不同数字来表示这个数字量的大小。这 10 个数字的每一个最多可以从 0 到 9 表示 10 个数字。如果想要表示一个比 9 大的数字量，可以使用两个或者更多的数字位，而每一个数字所处的位置表示了这一位的大小。例如，想要表示 23 这个数字量，可以使用(根据它们在该数字量中相应的位置)数字 2 表示 20 这个数字量，用数字 3 表示 3 这个数字量，如下所示。



◇ 十进制数字系统的基为 10。

十进制数中每一个数字所在的位置表示了这一位的大小，称之为权。整数的权是 10 的正次幂，从右向左递增，开始于 $10^0 = 1$。

$$\cdots 10^5\ 10^4\ 10^3\ 10^2\ 10^1\ 10^0$$

对于小数，权是 10 的负次幂，从左向右递减，开始于 10^{-1}。

$$10^2\ 10^1\ 10^0.10^{-1}\ 10^{-2}\ 10^{-3} \cdots$$

↑—— 小数点

◇ 数字的值取决于它在数中的位置。

十进制数的值等于每个数字乘以相应位置上的权之后的和，如例2.1和例2.2所示。

例2.1 把十进制数47表示为各位数字值的求和。

解： 数字4的权是10，由其所在的位置可知为10^1。数字7的权是1，由其所在的位置可知为10^0。

$$47 = (4 \times 10^1) + (7 \times 10^0)$$
$$= (4 \times 10) + (7 \times 1) = \mathbf{40 + 7}$$

相关问题①：确定939中每个数字的值。

例2.2 把十进制数568.23表示为各位数字值的求和。

解： 整数5的权是100，即10^2，整数6的权是10，即10^1，整数8的权是1，即10^0，小数2的权是0.1，即10^{-1}，小数3的权是0.01，即10^{-2}。

$$568.23 = (5 \times 10^2) + (6 \times 10^1) + (8 \times 10^0) + (2 \times 10^{-1}) + (3 \times 10^{-2})$$
$$= (5 \times 100) + (6 \times 10) + (8 \times 1) + (2 \times 0.1) + (3 \times 0.01)$$
$$= \mathbf{500} + \mathbf{60} + \mathbf{8} + \mathbf{0.2} + \mathbf{0.03}$$

相关问题：确定67.924中每个数字的值。

2.1节 温故而知新 (答案在本章的结尾。)

1. 下面的各个数中数字7的权是多少？
   (a)1370 (b)6725 (c)7051 (d)58.72
2. 对于下面每个十进制数，使用各位的数字乘以它相应的权然后求和的方法来表示。
   (a)51 (b)137 (c)1492 (d)106.58

2.2 二进制数

2.2.1 二进制计数

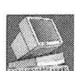
◇ 二进制数字系统有两个数字(位)。

下面学习二进制系统的计数方法。首先看一下十进制系统的计数方法。在用完数字之前，从0开始，依次计数到9。然后从另一个数位开始(左边)，继续从10计数到99。此时，已经用完了两位数字的所有组合，所以需要第三个数位，从100计数到999。

二进制数在计数时，也会发生类似的情况，只是这时只有两个数字，称为位(比特)。开始计数：0，1。此时，已经使用了两个数字，所以加入另一个数位继续计数：10，11。至此，已经使用了两个数字的所有组合，所以需要第3个数位。使用3个数位，可以继续计数：100，101，110，111。还要继续下去，就需要第4个数位，以此类推。从0到15的二进制计数方法如表2.1所示。注意在每一列中1和0的交替规律。

◇ 二进制数字系统的基为2。

① 答案在本章的结尾。



计算机小知识

计算机在运行时，有许多必须遇见的情况，即把 1 和一个存储在计数器中的数相加或相减。计算机具有一些特殊的指令，这些指令比加或减指令使用更少的时间，从而生成更少的机器代码。对于 Intel 微处理器，指令 INC 表示对一个数加 1。对于减法，相应的指令是 DEC，即对一个数减 1。

如表 2.1 所示，从 0 计数到 15 需要 4 个数位。一般来说，对于 n 个数位，可以计数到 $2^n - 1$。

$$最大十进制数 = 2^n - 1$$

例如，若有 5 个数位($n=5$)，就可以从 0 计数到 31。

$$2^5 - 1 = 32 - 1 = 31$$

若有 6 个数位($n=6$)，就可以从 0 计数到 63。

$$2^6 - 1 = 64 - 1 = 63$$

◇ 一个数中位的值由它在此数的位置确定。

表 2.1

十进制数	二进制数			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

2.2.2　二进制数的加权结构

◇ 位的权或值在二进制数中从右向左增加。

二进制数是有权数。在二进制整数中，最右边的位是最低有效位(LSB)，并且相应的权是 $2^0 = 1$。权从右向左，每前进一位，2 的幂次增加 1。最左边的位是最高有效位(MSB)。二进制数位的大小确定了它的权。

利用二进制也可以表示小数，在二进制小数点的右边添加相应的位就可以了，就像把十进制数位添加在十进制小数点的右边一样。在二进制小数中，最左边的位是最高有效位，其相应的权是 $2^{-1} = 0.5$。小数的权从左向右减少，每位相差 2 的 -1 次幂。二进制数的加权结构是

$$2^{n-1} \cdots 2^3\, 2^2\, 2^1\, 2^0 \,.\, 2^{-1}\, 2^{-2} \cdots 2^{-n}$$

↑—二进制小数点

其中 n 是从二进制小数点开始的位数。因此，在二进制小数点左边的所有位，其相应的权是 2 的正数幂，就像前面所讨论的整数一样。在二进制小数点右边的所有位，其相应的权是 2 的负数幂，或称小数权。

8 位二进制整数和 6 位二进制小数所对应的 2 的幂次，以及它们对应的十进制权，如表 2.2 所示。注意对于 2 的正数幂，权将增大至两倍，而对于 2 的负数幂，权将减半。通过倍增 2 的最高有效正数幂，并二等分 2 的最低有效负数幂，就可以很容易地扩展这个表。例如，$2^9 = 512$ 和 $2^{-7} = 0.0078125$。

表 2.2　二进制权

2的正次幂(正整数)									2的负次幂(小数)					
2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}
256	128	64	32	16	8	4	2	1	1/2	1/4	1/8	1/16	1/32	1/64
									0.5	0.25	0.125	0.625	0.03125	0.015625



计算机小知识

计算机使用二进制数来选择存储单元。每个单元都被赋予一个特定的数，称为地址。例如，一些微处理器具有32条地址线，可以选择2^{32} (4 294 967 296)个唯一的单元。

2.2.3 二进制数到十进制数的转换

◇ 把二进制数中所有位是1的权相加得到十进制数。

对于二进制数转换成十进制数，只要把二进制数中的所有位是1的权加起来，不考虑所有位是0的权。

例2.3 把二进制数1101101转换为十进制数。

解： 确定每个位是1的权，然后把这些权加起来得到十进制数。

$$
\begin{array}{rl}
\text{权：} & 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0 \\
\text{二进制数：} & 1\ \ 1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 1
\end{array}
$$

$$
\begin{aligned}
1101101 &= 2^6 + 2^5 + 2^3 + 2^2 + 2^0 \\
&= 64 + 32 + 8 + 4 + 1 = \mathbf{109}
\end{aligned}
$$

相关问题： 把二进制数10010001转换为十进制数。

例2.4 将二进制数0.1011转换为十进制数。

解： 确定每个位是1的权，然后将这些权加起来得到十进制数。

$$
\begin{array}{rl}
\text{权：} & 2^{-1}\ \ 2^{-2}\ \ 2^{-3}\ \ 2^{-4} \\
\text{二进制数：} & 0.1\ \ \ 0\ \ \ 1\ \ \ 1
\end{array}
$$

$$
\begin{aligned}
0.1011 &= 2^{-1} + 2^{-3} + 2^{-4} \\
&= 0.5 + 0.125 + 0.0625 = \mathbf{0.6875}
\end{aligned}
$$

相关问题： 将二进制数10.111转换为十进制数。

2.2节 温故而知新

1. 用8位二进制数表示的最大的十进制数是什么？
2. 二进制数10000中1的权。
3. 把二进制数10111101.011转换为十进制数。

2.3 十进制数到二进制数的转换

2.3.1 权和的方法

◇ 如果要把一个已知的十进制数转换为二进制数，只要求出二进制数的权，全部权加起来就是十进制数。

想要得到一个给定十进制数的二进制数，只要确定二进制数权的和，它们等于相应的十进制数。记住二进制权的简单方法是最低位为1，也就是2^0，任何一个权乘以2，就会得到下一个更高位的权，因此7个二进制权的序列就是64，32，16，8，4，2，1，这和上一节学到的一样。例如，十进制数9就可以由二进制权的和表示如下：

$$9 = 8 + 1 \text{ 或者 } 9 = 2^3 + 2^0$$

把1放在适当的权的位置上，即2^3和2^0，把0放在2^2和2^1的位置上，就确定了十进制数9。

$$
\begin{array}{cccc}
2^3 & 2^2 & 2^1 & 2^0 \\
1 & 0 & 0 & 1
\end{array}
\quad \text{十进制数9的二进制数表达}
$$

例 2.5　将下面的十进制数转换为二进制数：

(a)12　(b)25　(c)58　(d)82

解：(a)$12 = 8 + 4 = 2^3 + 2^2$ ⟶ **1100**

(b)$25 = 16 + 8 + 1 = 2^4 + 2^3 + 2^0$ ⟶ **11001**

(c)$58 = 32 + 16 + 8 + 2 = 2^5 + 2^4 + 2^3 + 2^1$ ⟶ **111010**

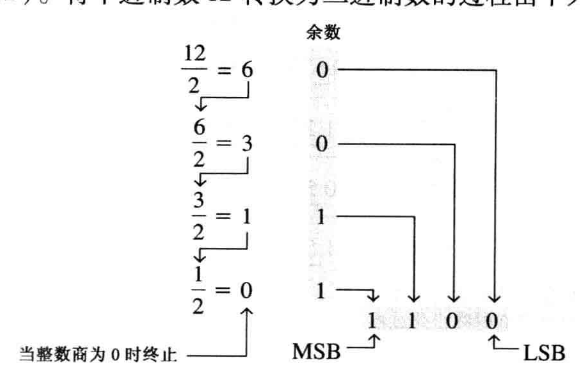
(d)$82 = 64 + 16 + 2 = 2^6 + 2^4 + 2^1$ ⟶ **1010010**

相关问题：将十进制数 125 转换为二进制数。

2.3.2　重复除以 2 的方法

◇ 要得到一个给定十进制数的二进制数，可以用 2 除这个十进制数直至商为 0，每次余数的全部便构成了二进制数。

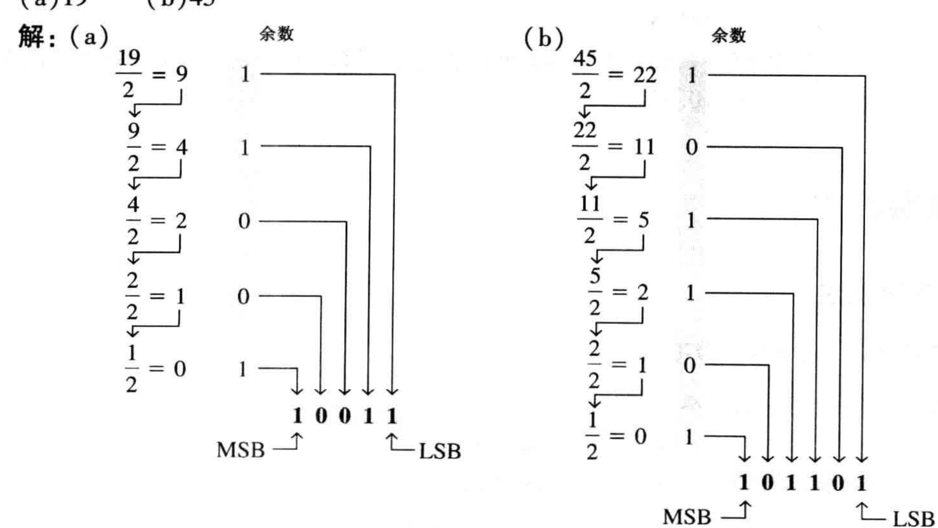
将十进制整数转换为二进制数的系统方法是重复除以 2 的过程。例如，把十进制数 12 转换为二进制数，首先 12 除以 2。然后把每次得到的商都除以 2，直到商为 0。每次相除所得到的余数就构成了二进制数。第一个得到的余数是二进制数中的最低有效位（LSB），最后一个产生的余数是最高有效位（MSB）。将十进制数 12 转换为二进制数的过程由下列步骤给出。



例 2.6　把下面的十进制数转换为二进制数。

(a)19　(b)45

解：



相关问题：将十进制数 39 转换为二进制数。

2.3.3 十进制小数转换为二进制数

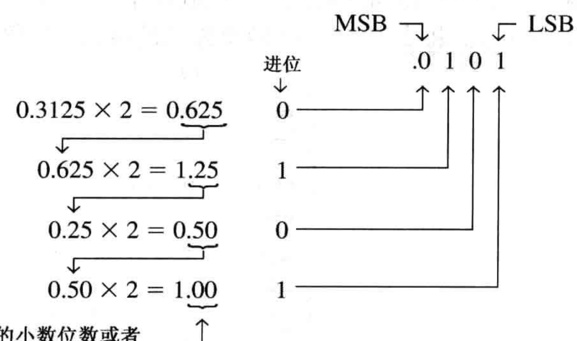
例2.5和例2.6给出了整数的转换，现在来看小数的转换。记住小数二进制权的一个简单的方法是最高有效权是0.5，也就是2^{-1}，任何一个权除以2，就得到次低位的权；因此4个小数的二进制权的序列就是0.5，0.25，0.125，0.0625。

权和 权和的方法可以应用于十进制小数，如以下的例子所示：

$$0.625 = 0.5 + 0.125 = 2^{-1} + 2^{-3} = 0.101$$

在2^{-1}位置上有一个1，在2^{-2}位置上有一个0，而在2^{-3}位置上有一个1。

重复乘2 正如所见，可以用重复除以2的方法把十进制整数转换为二进制数。二进制小数则可以用重复乘2的方法转换得到。例如，把十进制小数0.3125转换为二进制数，首先把0.3125乘以2，然后把每次乘积的小数部分乘以2，直到乘积的小数部分为0，或者达到了所需要的小数位数。由相乘产生的进位数字或者进位，就生成了二进制数。所产生的第一个进位是最高有效位(MSB)，最后一个进位是最低有效位(LSB)。该过程如下所示：



2.3节 温故而知新

1. 使用权和的方法，把下面每个十进制数转换为二进制数。
   (a)23 (b)57 (c)45.5
2. 使用重复除以2的方法(小数使用重复乘2的方法)，把下面每个十进制数转换成二进制数。
   (a)14 (b)21 (c)0.375

2.4 二进制算术

2.4.1 二进制加法

◇ 在二进制中，1+1=10，而不是2。

二进制数(位)加法的4条基本规则如下：

0+0=0 和为0，进位是0
0+1=1 和为1，进位是0
1+0=1 和为1，进位是0
1+1=10 和为10，进位是1

注意：前三条规则产生单个的位，而在第4条规则中，两个1相加生成二进制的2(10)。二进制数在相加时，最后一种情况在低位的和为0，而在左边的高位产生了进位1，如下面的加法运算11 +1所示：

```
进位  进位
 1 ←┐ 1 ←┐
 0  │ 1  │ 1
+0  │ 0  │ 1
────┼────┼───
 1  └ 0  └ 0
```

在最右列中，1 +1 =0，因而在此列的左侧一列，即中间一列产生进位1。在中间一列中，1 +1 +0 =0，因而在此列的左侧一列中产生进位1。在最左列中，1 +0 +0 =1。

当存在进位1时，就会遇到三个位进行加法的情况(两个数中的位和一个进位)。这种情况如下所示：

进位的位

1 + 0 + 0 = 01　　和为1，进位是0
1 + 1 + 0 = 10　　和为0，进位是1
1 + 0 + 1 = 10　　和为0，进位是1
1 + 1 + 1 = 11　　和为1，进位是1

例2.7　对下列二进制数进行加法运算：

(a)11 +11　(b)100 +10　(c)111 +11　(d)110 +100

解：同时给出相应的十进制加法作为对照。

```
(a)  11     3   (b) 100    4   (c) 111     7   (d)  110     6
    +11    +3       +10   +2       + 11   +3       +100    +4
    110     6       110    6       1010   10       1010    10
```

相关问题：把1111和1100相加。

2.4.2　二进制减法

◇ 在二进制中，10 -1 =1，而不是9。

二进制数(位)减法的4条基本规则如下：

0 -0 =0
1 -1 =0
1 -0 =1
10 -1 =1　0 -1产生借位1

进行减法运算时，有时必须从左边一列中借位。在二进制中，仅当0减去1时才需要借位。在这种情况下，从左边一列借来1，被减的列就会出现10，这时必须使用上面列出的4条基本规则中的最后一条。例2.8和例2.9说明了二进制减法，同时也给出了相应的十进制减法。

例2.8　完成下列二进制减法：

(a)11 -01　(b)11 -10

解：

```
(a)  11     3   (b)  11    3
    -01    -1       -10   -2
     10     2        01    1
```

在示例中不需要借位。二进制数01和1是一样的。

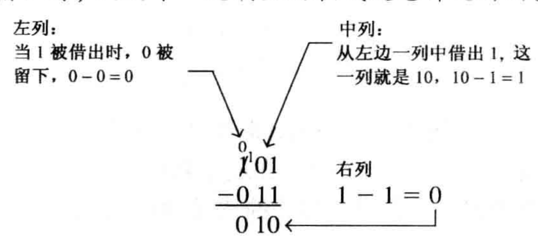
相关问题：从111中减去100。

例 2.9 从 101 中减去 011。

解：

$$\begin{array}{r} 101 \\ -011 \\ \hline \mathbf{010} \end{array} \qquad \begin{array}{r} 5 \\ -3 \\ \hline 2 \end{array}$$

仔细地看一看，在需要借位时，这两个二进制数的相减是怎样进行的。从右面一列开始。



相关问题：从 110 中减去 101。

2.4.3 二进制乘法

◇ 两位的二进制乘法和十进制数 0 和 1 的乘法相同。

位相乘的 4 条基本规则如下：

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

二进制乘法和十进制乘法的运算方法是一样的。这涉及部分积的形成，把相继的部分积向左移一位，然后把所有的部分积加起来。例 2.10 说明了这个过程，同时也给出了相应的十进制乘法作为参照。

例 2.10 计算下面的二进制乘法：

(a)11 × 11 (b)101 × 111

解：(a)

$$\begin{array}{r} 11 \\ \times 11 \\ \hline 11 \\ +11\ \\ \hline \mathbf{1001} \end{array} \qquad \begin{array}{r} 3 \\ \times 3 \\ \hline 9 \end{array}$$

部分积：11，+11

(b)

$$\begin{array}{r} 111 \\ \times 101 \\ \hline 111 \\ 000\ \\ +111\ \ \\ \hline \mathbf{100011} \end{array} \qquad \begin{array}{r} 7 \\ \times 5 \\ \hline 35 \end{array}$$

部分积：111，000，+111

相关问题：1101 × 1010。

2.4.4 二进制除法

◇ 一个计数器可以用来进行二进制的算术运算，只要不超出计数器的运算范围。

二进制中的除法遵循和十进制除法一样的过程，如例 2.11 所示，其中给出了相应的十进制除法。

例 2.11 执行下面的二进制除法运算：

(a)110 ÷ 11 (b)110 ÷ 10

解：(a)

$$\begin{array}{r} \mathbf{10} \\ 11\overline{)110} \\ \underline{11}\ \\ 000 \end{array} \qquad \begin{array}{r} 2 \\ 3\overline{)6} \\ \underline{6} \\ 0 \end{array}$$

(b)

$$\begin{array}{r} \mathbf{11} \\ 10\overline{)110} \\ \underline{10}\ \\ 10 \\ \underline{10} \\ 00 \end{array} \qquad \begin{array}{r} 3 \\ 2\overline{)6} \\ \underline{6} \\ 0 \end{array}$$

相关问题： 1100 除以 100。

2.4 节　温故而知新

1. 完成下面的二进制加法：
   (a) 1101 + 1010　(b) 10111 + 01101
2. 完成下面的二进制减法：
   (a) 1101 − 0100　(b) 1001 − 0111
3. 完成下面的二进制运算：
   (a) 110 × 111　(b) 1100 ÷ 011

2.5　二进制数的反码和补码

2.5.1　求二进制数的反码

◇ 改变数中的每一位以得到反码。

二进制数的反码可以通过把所有的 1 变为 0 及把所有的 0 变为 1 而得到，如下所示：

1 0 1 1 0 0 1 0　二进制数
↓↓↓↓↓↓↓↓
0 1 0 0 1 1 0 1　反码

利用数字电路求二进制数反码的最简单方法是使用并行反相器（“非”电路），如图 2.1 所示，用以变换 8 位二进制数。

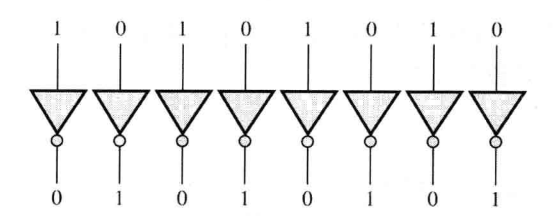


图 2.1　反相器的例子，用以获得二进制数的反码

2.5.2　求二进制数的补码

◇ 反码加 1 就可以得到补码。

二进制数的补码是通过在反码的最低有效位（LSB）上加 1 而获得的。

$$补码 = 反码 + 1$$

例 2.12　求 10110010 的补码。

解：

10110010	二进制数
01001101	反码
+ 1	加 1
01001110	补码

相关问题： 确定 11001011 的补码。

◇ 改变最低有效 1 左边的全部位求得补码。

求二进制数补码的另一种替代方法如下所示：

1. 从右边的最低有效位开始，写下它们实际的位，包括第一个 1。
2. 剩下的位求反码。

例 2.13 使用替代方法求 10111000 的补码。

解：

10111000 二进制数
01001000 补码

原数位的反码（0100）；这些位保持不变（1000）

相关问题： 求 11000000 的补码。

使用反相器和加法器可以实现二进制负数的补码，如图 2.2 所示。该图说明了怎样把一个 8 位数转换为它的补码；首先每一个位取反(取得反码)，然后使用加法器电路把反码加 1。

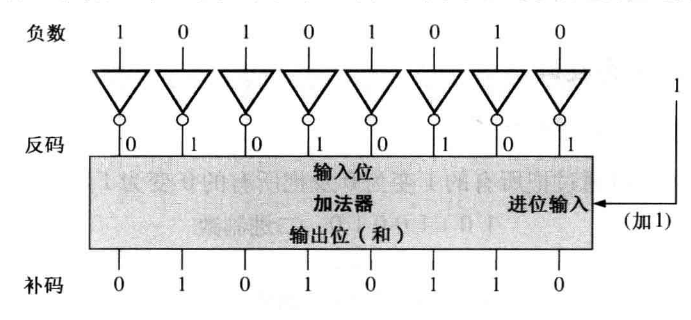


图 2.2 获得二进制负数补码的例子

为了把反码或者补码变回二进制原码(非补码)，可以应用和前面描述的一样的过程。为了从反码回到二进制数原码，可以反转所有的位。为了从补码回到二进制数原码，可以先取得补码的反码，然后在最低有效位上加 1。

2.5 节 温故而知新

1. 求出下面每个二进制数的反码：
   (a)00011010 (b)11110111 (c)10001101
2. 求出下面每个二进制数的补码：
   (a)00010110 (b)11111100 (c)10010001

2.6 带符号数

2.6.1 符号位

二进制符号数的最左边就是符号位，指出这个数是正数还是负数。

符号位 0 表示正数，1 表示负数。

2.6.2 符号数值的形式

当以符号数值的形式表示带符号二进制数时，最左边的一位是符号位，其余的都是数值位。数值位对于正数和负数来说都是二进制原码(非补码)。例如，十进制数 +25 表示为符号数值的

形式就是一个 8 位带符号二进制数，使用符号数值的形式为

00011001

符号位 ↑ ↑ 数值位

十进制数 -25 表示为

10011001

注意 +25 和 -25 之间的唯一区别是符号位，因为对于正数和负数来说，数值位都是二进制原码。

在符号数值的形式中，负数和其相应的正数具有相同的数值位，但其符号位为 1 而不是 0。



计算机小知识

计算机在所有的算术运算中都使用补码来表示负整数。原因是减去某个数和加上这个数的补码是一样的。计算机通过按位取反和加 1 来形成补码，使用特殊的指令产生和图 2.2 中的加法器一样的结果。

2.6.3 反码形式

以反码形式表示正数的方法和以符号数值形式表示正数的方法是一致的。但是，负数却是其相应正数的反码。例如，使用 8 位数字，十进制数 -25 可以表示为 +25(00011001)的反码：

11100110

在反码形式中，负数就是其相应正数的反码。

2.6.4 补码形式

正数的补码形式表示方法和符号数值与反码形式的表示方法是一致的。负数是相应正数的补码。再次使用 8 位数字，把十进制数 -25 表示为 +25(000011001)的补码。按位取反再加 1，得到

$$-25 = 11100111$$

在补码形式中，负数是相应正数的补码。

例 2.14 以符号数值、反码和补码的形式将十进制数 -39 表示为一个 8 位数。

解： 首先，写出 +39 的 8 位数。

00100111

在符号数值的形式中，-39 是通过改变符号位为 1 并保持数值位不变而得到的。这个数是

10100111

在反码形式中，-39 是通过求 +39 (00100111)的反码得到的，如下所示：

11011000

在补码形式中，-39 是通过求 +39 (00100111)的补码得到的，如下所示：

```
  11011000    反码
+        1
----------
  11011001    补码
```

相关问题： 以符号数值、反码和补码的形式表示 +19 和 -19。

2.6.5 带符号数的十进制值

符号数值 在符号数值的形式中，正数和负数的十进制值，是由所有数值位为1的相应权加起来得到的，不考虑那些为0的位。符号通过检查符号位来确定。

例2.15 确定以符号数值表示的带符号二进制数的十进制值：10010101。

解： 这7个数值位和它们以2的幂表示的权如下所示：

2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	1	0	1	0	1

把1位置上所对应的权加起来，

$$16+4+1=21$$

符号位为1，所以十进制数是 **−21**。

相关问题： 确定符号数值表示的数01110111的十进制值。

反码 在反码形式中，正数的十进制值是由所有为1的数值位相应的权加起来得到的，而不要考虑为0的位置。负数的十进制值是通过给符号位的权赋以负值，并把所有为1的数值位相应的权加起来再加上1得到的。

例2.16 确定以反码表示的带符号二进制数的十进制值：

(a)00010111　　(b)11101000

解： (a)正数的位和它们以2的幂表示的权如下所示：

-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	1	0	1	1	1

把所有位置为1的权加起来，

$$16+4+2+1=\mathbf{+23}$$

(b)负数的位和它们以2的幂表示的权如下所示。注意负符号位的权是 -2^7 或者 −128。

-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	1	0	1	0	0	0

把所有位置为1的权加起来，

$$-128+64+32+8=-24$$

把结果加上1，最终的十进制数是

$$-24+1=\mathbf{-23}$$

相关问题： 确定反码数11101011的十进制值。

补码 在补码形式中，正数和负数的十进制值，是把所有为1的数值位相应的权加起来得到的，而不要考虑为0的位置。负数中符号位的权被赋予负值。

例2.17 确定以补码表示的带符号二进制数的十进制值：

(a)01010110　　(b)10101010

解： (a)正数的位和它们以2的幂表示的权如下所示：

-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	0	1	0	1	1	0

把所有位置为 1 的权加起来，

$$64 + 16 + 4 + 2 = \mathbf{+86}$$

(b)负数的位和它们以 2 的幂表示的权如下所示。注意负符号位的权是 -2^7 或者 -128。

-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	0	1	0	1	0	1	0

把所有位置为 1 的权加起来，

$$-128 + 32 + 8 + 2 = \mathbf{-86}$$

相关问题：确定补码数 11010111 的十进制值。

从这些例子中，可以看到为什么用补码形式来表示带符号整数比较好：在转换为十进制数时，仅仅需要求权的和，而不用考虑是正数还是负数。对于负数而不是正数，反码形式需要给权的和再加上 1。当然反码形式并不是常用的，因为 0 的反码有两种可能的表示方式(00000000 或 11111111)。

2.6.6　带符号整数的表示范围

◇ 二进制数的数值范围取决于数的位数(n)。

使用 8 位数给出说明，因为在大多数计算机中通常使用 8 位的组合，并被赋予特殊的名称，即字节(byte)。使用一个字节或者 8 位，可以表达 256 个不同的数。使用两个字节或者16 位，可以表达 65 536 个不同的数。使用 4 个字节或者 32 位，可以表达 4.295×10^9 个不同的数。求解 n 位的不同组合个数的公式是

$$\text{总组合数} = 2^n$$

对于补码带符号数，n 位数的数值范围是

$$-(2^{n-1}) \text{ 到 } +(2^{n-1}-1)$$

其中对于每一个数，都有一个符号位和 $n-1$ 个数字位。例如，使用 4 位数，并以补码表示的数的范围是从 $-(2^3) = -8$ 到 $2^3 - 1 = +7$。类似地，利用 8 位，就可以从 -128 到 $+127$；使用 16 位，就可以从 $-32\ 768$ 到 $+32\ 767$；等等。

2.6.7　浮点数

为了表示很大的整数，就会需要多个位。当需要表示的数值同时具有整数和小数部分时(比如 23.5618)，就会有一个问题。基于科学计数法的浮点计数方法，可以表示很大及很小的数，而不用增加位数，当然也可以表示同时具有整数和小数部分的数。

浮点数(也称为实数)由两部分组成再加上一个符号。尾数(mantissa)是浮点数中用以表示数字数值的部分，大小在 0 和 1 之间。指数(exponent)是浮点数中用以表示小数点(或者二进制小数点)要移动的位数的部分。

十进制数的例子将有助于理解浮点数的基本概念。考虑一个十进制数，以整数形式表示为 241506800。尾数为.2415068，而指数是 9。当把这个整数表示为浮点数时，通过把小数点移动到数字的最左边而将其标准化，使得位数是一个小数，而指数是 10 的幂。这个浮点数写为

$$0.2415068 \times 10^9$$

对于二进制浮点数来说，其格式由 ANSI/IEEE 标准 754-1985 定义为三种形式：单精度、双精度及扩展精度。除了位数不同之外，它们都具有相同的基本格式。单精度浮点数具有 32 位，

双精度浮点数具有64位，而扩展浮点数具有80位。我们的讨论将限于单精度浮点数格式。



计算机小知识

在CPU(中央处理器)之外，计算机使用协处理器依靠浮点数来执行复杂的数学计算。目的是释放CPU资源去完成其他任务，从而提高性能。数字协处理器也称为浮点单元(FPU)。

单精度浮点二进制数 在单精度浮点二进制数的标准格式中，符号位(S)是最左边的位，指数(E)包括了接下来的8位，尾数或者小数部分(F)包括了剩余的23位，如下所示：

32位		
符号位(S)	指数(E)	尾数(小数，F)
1位	8位	23位

在尾数或者小数部分，一般认为二进制小数点位于23位的左边。为了有效起见，使尾数部分具有24位，因为在任何一个二进制数中最左边(MSB)总是为1。所以，认为这个1位于最左边，尽管并没有占用实际的位置。

指数中的8位表示的是偏移指数，在实际指数上加上127就可以得到。偏移指数的目的是允许表示很大或者很小的数，而不需要为指数赋予一个单独的符号位。偏移指数允许的实际指数范围是-126到+128。

为了说明二进制数怎样以浮点格式表示，使用1011010010001作为一个例子。首先，它可以表示为1加上一个二进制小数，把二进制小数点向左移动12位然后乘以适当的2的幂。

$$1011010010001 = 1.011010010001 \times 2^{12}$$

假设这是一个正数，符号位(S)是0。指数12表示为偏移指数的形式，将其加上127($12+127=139$)。该偏移指数(E)表示为二进制数10001011。尾数是二进制数的小数部分(F)，即.011010010001。因为在2的幂次表达式中，二进制小数点的左边总是为1，它并不包含在尾数中。完整的浮点数是

S	E	F
0	10001011	01101001000100000000000

接下来，看看怎样计算以浮点格式表示的二进制数。确定浮点数数值的一般方法如下面的公式所示：

$$数 = (-1)^{S}(1+F)(2^{E-127})$$

为了说明这个公式，考虑下面的浮点二进制数：

S	E	F
1	10010001	10001110001000000000000

符号位是1。偏移指数是10010001=145。应用这个公式，我们得到

$$\begin{aligned}数 &= (-1)^{1}(1.10001110001)(2^{145-127})\\ &= (-1)(1.10001110001)(2^{18}) = -1100011100010000000\end{aligned}$$

这个浮点二进制数等于十进制数的-407688。因为指数可以是-126和+128之间的任意数，所以最大和最小的数都可以表示出来。32位的浮点数可以代替具有129位的二进制整数。因为指数确定了二进制小数点的位置，所以也可以表示同时包含整数和小数部分的数。

浮点数的这种格式有两个例外：数0.0由全0来表示，而无穷大的数由指数全1和尾数全0来表示。

例 2.18 把十进制数 3.248×10^4 转换为单精度浮点二进制数。

解： 把十进制数转换为二进制数。

$$3.248 \times 10^4 = 32480 = 111111011100000_2 = 1.11111011100000 \times 2^{14}$$

最高有效位(MSB)并不占用位的位置，因为它总是为 1，所以尾数是 23 位的二进制小数 11111011100000000000000，而偏移量是

$$14 + 127 = 141 = 10001101_2$$

完整的浮点数是

0	10001101	11111011100000000000000

相关问题： 确定浮点二进制数的二进制值：01001100010000100010100110000000。

2.6 节 温故而知新

1. 以 8 位二进制符号数的形式表示十进制数 +9。
2. 以 8 位二进制反码数的形式表示十进制数 -33。
3. 以 8 位二进制补码数的形式表示十进制数 -46。
4. 列出浮点数的 3 个部分。

2.7 带符号数的算术运算

2.7.1 加法

加法中的两个数就是加数和被加数，结果是和。当两个带符号二进制数相加时，有以下 4 种情况。

1. 两个数都是正的。
2. 正数的数值大于负数的数值。
3. 负数的数值大于正数的数值。
4. 两个数都是负的。

这里使用 8 位带符号数，每次给出一种情况作为例子，也给出相应的十进制数作为对照。

◇ 两个正数相加产生一个正数。

两个数都是正的：

```
  00000111      7
+ 00000100    + 4
  00001011     11
```

和是正的，因而是二进制原码(非补码)。

◇ 正数和负数相加产生一个正数。

正数的数值大于负数的数值：

```
              00001111     15
            + 11111010   + -6
放弃进位——→ 1 00001001      9
```

最后的进位被舍去。和是正的，因而是二进制原码(非补码)。

◇ 一个正数加上一个较大的负数或者两个负数相加时，生成一个补码形式的负数。

负数的数值大于正数的数值：

```
   00010000        16
 + 11101000    + -24
   11111000        -8
```

和是负的，所以是补码形式。

两个数都是负的：

```
                      11111011      -5
                    + 11110111    + -9
放弃进位———————→  1  11110010     -14
```

最后的进位被舍去。和是负的，所以是补码形式。

在计算机中，负数是以补码形式保存的，所以正如所见，加法过程是很简单：将两个数加起来，并舍去最后的任何进位。

溢出条件 当两个数加在一起，而表示和所需的位数超出了这两个数的位数，这时就会发生溢出，并由一个错误符号位指明。溢出仅发生在两个都是正数或者两个都是负数的情况下。如果相加结果得到的符号位和相加的两个数的符号不同，就表明发生了溢出。下面的8位数例子将说明这种情况。

```
   01111101      125
 + 00111010    +  58
   10110111      183
```

符号不正确（指向最高位 1）

大小不正确（指向其余 7 位 0110111）

在这个例子中，和183需要8个数值位。由于数中只有7个数值位(有一位是符号位)，这时进位就会进入符号位，从而产生溢出指示。

一次加两个数 现在让我们看看数字串相加的情况，即一次加两个数。可以这样操作，首先将前两个数相加，然后两数之和加第三个数，然后再在此和的基础上加第四个数，以此类推。这就是计算机中数字串的相加方法。一次加两个数的方法如例2.19所示。

例2.19 符号数相加：01000100，00011011，00001110，00010010。

解： 给出相应的十进制加法作为对照。

十进制	二进制	
68	01000100	
+ 27	+ 00011011	加前两个数
95	01011111	第一个和
+ 14	+ 00001110	加第三个数
109	01101101	第二个和
+ 18	+ 00010010	加第四个数
127	**01111111**	最后的和

相关问题： 将00110011、10111111和01100011加起来。这些是带符号数。

2.7.2 减法

◇ 减法是将减数符号改变后的加法。

减法是加法的一个特例。例如，从 +9（被减数）中减去 +6（减数）就相当于 +9 加上 -6。基本上，减法运算是改变减数的符号然后加上被减数的运算。减法的结果称为差。

正二进制数或者负二进制数的符号通过求此数的补码而改变。

例如，求取正数 00000100（+4）的补码时，就会得到 11111100，这就是 -4，权和的计算如下：

$$-128+64+32+16+8+4=-4$$

作为另一个例子，当取负数 11101101（-19）的补码时，就会得到 00010011，这就是 +19，权和的计算如下：

$$16+2+1=19$$

◇ 当用补码的方法进行二进制减法时，重要的是两个数要有相同的位数。

由于减法仅仅是减数符号改变后的加法，所以该过程有如下的表述方式：

要将两个带符号数相减，取减数的补码然后相加即可。舍去最后的任何进位。

例 2.20 叙述了减法过程。

例 2.20　完成下面带符号数的减法运算：

(a)00001000 - 00000011　　(b)00001100 - 11110111

(c)11100111 - 00010011　　(d)10001000 - 11100010

解： 像其他例子一样，也给出了相应的十进制减法作为对照。

(a)在这种情况下，8 - 3 = 8 + (-3) = 5。

	00001000	被减数（+8）
	+ 11111101	减数（-3）的补码
舍去进位 ⟶	**1 00000101**	差（+5）

(b)在这种情况下，12 - (-9) = 12 + 9 = 21。

00001100	被减数(+12)
+ 00001001	减数(+9)的补码
00010101	差(+21)

(c)在这种情况下，-25 - (+19) = -25 + (-19) = -44

	11100111	被减数(-25)
	+ 11101101	减数(-19)的补码
舍去进位 ⟶	**1 11010100**	差(-44)

(d)在这种情况下，-120 - (-30) = -120 + 30 = -90

10001000	被减数(-120)
+ 00011110	减数(+30)的补码
10100110	差(-90)

相关问题： 01011000 减去 01000111。

2.7.3　乘法

乘法中的数分别是被乘数、乘数及积。下面的十进制乘法运算说明了这些数：

8	被乘数
× 3	乘数
24	积

在大多数计算机中，乘法运算是通过加法来完成的。正如所见，减法是由加法器完成的，现在看看乘法运算是如何完成的。

◇ 乘法相当于一个数加上它本身，相加的次数就是乘数。

直接加法和部分积是使用加法完成乘法的两种基本方法。在直接加法的方法中，被乘数自身相加的次数等于乘数。在前面的十进制例子(3×8)中，三个被乘数相加：8+8+8=24。这种方法的缺点是，如果乘数很大，则运算会变得很冗长。例如，要运算350×75，必须把350自身相加75次。顺便说一下，这就是为什么相乘的次数常用来指乘数。

两个二进制数相乘时，这两个数都必须是原码(非补码)的形式。直接加法的方法如例2.21所示，每次进行两个二进制数的相加。

例2.21 把带符号二进制数相乘：01001101(被乘数)和00000100(乘数)。使用直接加法的方法。

解： 由于两个数都是正数，所以都是原码，积就是正数。乘数的十进制数值为4，所以被乘数要自身相加4次，如下所示：

01001101	第一次
+ 01001101	第二次
10011010	部分和
+ 01001101	第三次
11100111	部分和
+ 01001101	第四次
100110100	积

由于被乘数的符号位是0，所以对结果没有影响。积中的所有位都是表示数值的大小。

相关问题： 使用直接加法的方法，把01100001乘以00000110。

部分积方法可能是最常用的一种方法，因为它反映了普通的手工乘法运算。从乘数的最低有效位开始，将被乘数乘以乘数的每一个位。被乘数乘以乘数每一位的结果称为部分积。每一个相继的部分积都向左移动(平移)一位，当产生所有的部分积时，把它们加起来就得到最后的积。这里有一个十进制例子。

239	被乘数
× 123	乘数
717	第一个部分积(3×239)
478	第二个部分积(2×239)
+ 239	第三个部分积(1×239)
29 397	最后的积

乘法运算的积的符号取决于被乘数和乘数的符号，依据下面两条规则：

- **如果符号相同，积就是正值；**
- **如果符号不同，积就是负值。**

二进制乘法的部分积方法的基本步骤如下所示。

步骤1： 确定被乘数和乘数的符号是相同的还是相异的，这将决定积的符号。

步骤2： 把所有的负数变为原码(非补码)形式。因为大多数计算机都以补码形式保存负数，将负数变换为原码，需要进行补码运算。

步骤3： 开始于最低有效乘数位，生成部分积。当乘数的位是1时，部分积与被乘数就是一样的。当乘数的位是0时，部分积就是0。将每一个相继的部分积向左移动一位。

步骤 4：将相继的部分积与其前面部分积的和相加，从而得到最终的积。

步骤 5：如果步骤 1 所确定的符号位是负的，就对积取补码。如果是正的，积就保持为原码。

例 2.22 把带符号二进制数相乘：01010011（被乘数）和 11000101（乘数）。

解：步骤 1：被乘数的符号位是 0，而乘数的符号位是 1。积的符号位将是 1（负数）。

步骤 2：取乘数的补码，把它变为原码。

11000101 ⟶ 00111011

步骤 3 和步骤 4：相乘过程如下所示。注意在这些步骤中只使用了数值位。

```
         1010011      被乘数
       × 0111011      乘数
         1010011      第一个部分积
      + 1010011       第二个部分积
        11111001      第一个和第二个部分积的和
     + 0000000        第三个部分积
       011111001      和
    + 1010011         第四个部分积
      1110010001      和
   +1010011           第五个部分积
     100011000001     和
  + 1010011           第六个部分积
    1001100100001     和
 + 0000000            第七个部分积
   1001100100001      最后的积
```

步骤 5：由于积的符号是 1，正如步骤 1 中所确定的那样，因此求积的补码。

1001100100001 ⟶ 0110011011111

补充符号位

└⟶ **1 0110011011111**

相关问题：通过变换成十进制数并完成乘法运算，证明以上相乘过程是正确的。

2.7.4 除法

除法中的数分别是被除数、除数及商。下面给出了标准的除法格式。

$$\frac{\text{被除数}}{\text{除数}} = \text{商}$$

计算机中的除法运算是通过减法完成的。由于减法是由加法器完成的，所以除法也可以通过加法器来完成。

除法的结果称为商；商是除数可以进入被除数的次数。这就是被除数中可以减去多少次除数的次数等于商，如下面的 21 除以 7 所示。

```
  21    被除数
 − 7    第一次减去除数
  14    第一个部分余数
 − 7    第二次减去除数
   7    第二个部分余数
 − 7    第三次减去除数
   0    余数为 0
```

在这个简单的例子中，在余数为 0 之前除数从被除数中减去了三次，所以商是 3。

商的符号取决于被除数和除数的符号，依据以下两条规则：

- **如果符号相同，商就是正的。**
- **如果符号不同，商就是负的。**

当两个二进制数相除时，这两个数都必须是原码(非补码)形式。除法的基本过程如下。

步骤1： 确定被除数和除数的符号是相同还是不同，这将确定商的符号。商的初始值是0。

步骤2： 使用补码加法把除数从被除数中减去，得到第一个部分余数，同时将商加1。如果这个部分余数是正的，转到步骤3。如果部分余数是0或者是负的，就完成了除法。

步骤3： 从部分余数中减去除数，商加上1。如果结果是正的，重复以上步骤得到下一个部分余数。如果结果是零或负的，完成除法。

继续从被除数和部分余数中减去除数，直至出现0或者负数结果。计算除数被减的次数，就会得到商。例2.23使用8位带符号二进制数给出了这些步骤。

例2.23 01100100除以00011001。

解：步骤1： 这两个数的符号都是正的，所以商也是正的。商的初值为0：00000000。

步骤2： 使用补码加法从被除数中减去除数(记住舍去最后的进位)。

01100100	被除数
+ 11100111	除数的补码
01001011	第一个正的部分余数

商+1：00000000+00000001=00000001。

步骤3： 使用补码加法从第一个部分余数中减去除数。

01001011	第一个部分余数
+ 11100111	除数的补码
00110010	第二个正的部分余数

商+1：00000001+00000001=00000010。

步骤4： 使用补码加法从第二个部分余数中减去除数。

00110010	第二个部分余数
+ 11100111	除数的补码
00011001	第三个正的部分余数

商加1：00000010+00000001=00000011。

步骤5： 使用补码加法从第三个部分余数中减去除数。

00011001	第三个部分余数
+ 11100111	除数的补码
00000000	余数0

商加1：00000011+00000001=**00000100**(最终的商)，计算过程就完成了。

相关问题： 转换为十进制数并完成该除法运算，证明以上过程是正确的。

2.7节 温故而知新

1. 列出加法的4种情况。
2. 把符号数00100001和10111100相加。
3. 把符号数01110111和00110010相减。
4. 当两个负数相乘时，积的符号是什么？
5. 01111111乘以00000101。
6. 当一个正数除以一个负数时，商的符号是什么？
7. 计算00110000除以00001100。

2.8　十六进制数

◇ 十六进制数字系统由数字 0 ~ 9 和字母 A ~ F 组成。

十六进制数字系统的基是 16，也就是它由 16 个数字和字母共同组成。大多数的数字系统都能成组处理二进制数，即多个 4 位的组合，这样使用十六进制非常方便，因为每一个十六进制数位就代表了一个 4 位二进制数(如表 2.3 所示)。

10 个数字和 6 个字母字符构成了十六进制数字系统。使用字母 A、B、C、D、E、F 表示数字，初看起来很奇怪，但是请记住任何一种数字系统都是有序符号的集合而已。如果已经理解这些符号表示哪些数，一旦习惯了它们的使用方式，这些符号本身就不重要了。我们将使用下标 16 来指明十六进制数以避免和十进制数相混淆。有时可能会看到十六进制数后面跟随一个“h”。

表 2.3

十进制数	二进制数	十六进制数
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F



计算机小知识

对于千兆字节计的计算机存储器，以二进制指定存储地址是十分烦琐的。例如，在一个 4 GB 的存储器中，指定一个地址就需要 32 个数位。而使用 8 个十六进制数位来表示 32 位编码就要方便多了。

2.8.1　十六进制计数

计数到 F 时，怎样在十六进制中计数呢？只要到左侧一列继续计数就可以了，如下所示：

…, E, F, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1A, 1B, 1C, 1D, 1E, 1F,
20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2A, 2B, 2C, 2D, 2E, 2F, 30, 31,…

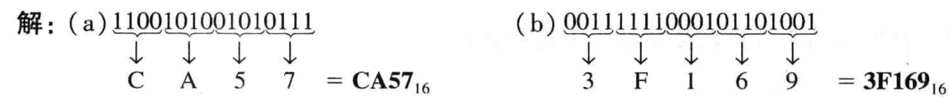
利用两个十六进制数位，可以数到 FF_{16}，也就是十进制数中的 255。当大于 255 时，就需要三个十六进制数位。例如，100_{16} 在十进制数中就是 256，101_{16} 在十进制数中就是 257，以此类推。最大的 3 位十六进制数是 FFF_{16}，表示十进制数 4095。最大的 4 位十六进制数是 $FFFF_{16}$，也就是十进制数 65 535。

2.8.2　二进制数到十六进制数的转换

把二进制数转换为十六进制数是非常直接的过程。从最右边一位开始，将二进制数每 4 位分成一组，用对等的十六进制符号替代相应的每个 4 位的组合。

例 2.24　把下面的二进制数转换为十六进制数。

(a) 1100101001010111　(b) 111111000101101001



在(b)部分中添加了两个0，以形成左边的4位的组合。

相关问题：将二进制数1001111011110011100转换为十六进制数。

2.8.3 十六进制数到二进制数的转换

◇ 使用十六进制数来表示二进制数是一个很方便的方法。

把十六进制数转换为二进制数的过程和上述过程相反，使用对等的4位的组合替代每个十六进制符号。

例2.25 确定下面的十六进制数的二进制数。

(a)$10A4_{16}$　(b)$CF8E_{16}$　(c)9742_{16}

解：(a) 1 0 A 4 → **1000010100100**　(b) C F 8 E → **1100111110001110**　(c) 9 7 4 2 → **1001011101000010**

在(a)部分中，认为最高有效位的前面还有3个0，因而构成了一个4位的组合。

相关问题：将十六进制数6BD3转换为二进制数。

◇ 十六进制数和二进制数之间的转换直接且容易。

应该清楚，处理十六进制数要比处理相应的二进制数容易一些。因为它们之间的转换很容易，所以十六进制系统在程序设计、打印输出及显示中得到了广泛的应用。

2.8.4 十六进制数到十进制数的转换

一种求十六进制数相应的十进制数的方法是，首先把十六进制数转换为二进制数，然后再把二进制数转换为十进制数。

例2.26 把下面的十六进制数转换为十进制数：

(a)$1C_{16}$　(b)$A85_{16}$

解：记住，首先将十六进制数转换为二进制数，然后转换为十进制数。

(a) 1 C

$$\underbrace{0001}\underbrace{1100} = 2^4 + 2^3 + 2^2 = 16 + 8 + 4 = \mathbf{28_{10}}$$

(b) A 8 5

$$\underbrace{1010}\underbrace{1000}\underbrace{0101} = 2^{11} + 2^9 + 2^7 + 2^2 + 2^0 = 2048 + 512 + 128 + 4 + 1 = \mathbf{2693_{10}}$$

相关问题：将十六进制数6BD转换为十进制数。

◇ 一台计数器可以用来完成十六进制的算术运算。

另一种把十六进制数转换为十进制数的方法是，把每一个十六进制数位的十进制值都乘以该位的权，然后再把这些积加起来。十六进制数的权是16的递增幂(从右到左)。对于4位十六进制数，它的权是

16^3	16^2	16^1	16^0
4096	256	16	1

例2.27 把下面的十六进制数转换为十进制数：

(a)$E5_{16}$　(b)$B2F8_{16}$

解： 回忆表 2.3，其中字母 A 到 F 分别表示十进制数 10 到 15。

(a) $E5_{16} = (E \times 16) + (5 \times 1) = (14 \times 16) + (5 \times 1) = 224 + 5 = \mathbf{229_{10}}$

(b)
$$\begin{aligned} B2F8_{16} &= (B \times 4096) + (2 \times 256) + (F \times 16) + (8 \times 1) \\ &= (11 \times 4096) + (2 \times 256) + (15 \times 16) + (8 \times 1) \\ &= 45\,056 + 512 + 240 + 8 = \mathbf{45\,816_{10}} \end{aligned}$$

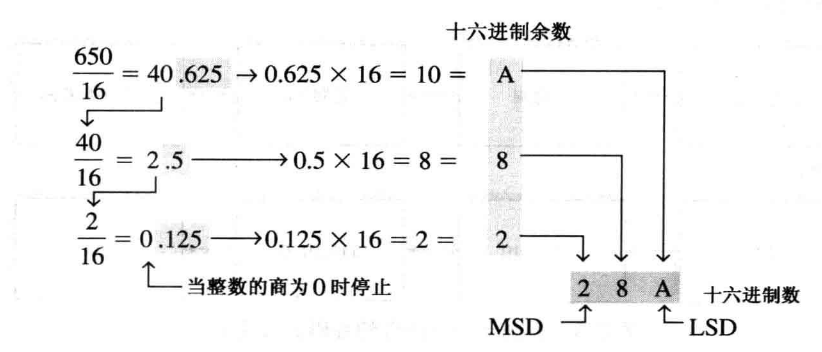
相关问题： 把 $60A_{16}$ 转换为十进制数。

2.8.5 十进制数到十六进制数的转换

用 16 重复除十进制数，就会生成相应的十六进制数，它由相除的余数构成。生成的第一个余数是最低有效数(LSD)。以 16 作为除数的每一次相除都会生成一个余数，这个余数就是相应十六进制数的一个数位。这个过程和 2.3 节提到的十进制到二进制转换中重复除以 2 的方法相似。例 2.28 展示了这个过程。注意当商有小数部分时，用除数乘小数部分就会得到余数。

例 2.28 用重复除 16 的方法把十进制数 650 转换为十六进制数。

解：



相关问题： 把十进制数 2591 转换为十六进制数。

2.8.6 十六进制加法

可以直接使用十六进制数进行加法运算，记住十六进制数 0 ~ 9 等同于十进制数 0 ~ 9，而十六进制数 A ~ F 等同于十进制数 10 ~ 15。当两个十六进制数相加时，使用下面的方法。(十进制数由下标 10 表示。)

1. 在加法问题任一给定的列中，把两个十六进制数看成它们的十进制值。例如，$5_{16} = 5_{10}$ 和 $C_{16} = 12_{10}$。
2. 如果这两个数字的和是 15_{10} 或者小些，记下相应的十六进制数。
3. 如果这两个数字的和大于 15_{10}，记下超出 16_{10} 的量，并在下一列进 1。

例 2.29 把下面的十六进制数相加：

(a) $23_{16} + 16_{16}$　(b) $58_{16} + 22_{16}$　(c) $2B_{16} + 84_{16}$　(d) $DF_{16} + AC_{16}$

解： (a)
$$\begin{array}{r} 23_{16} \\ +16_{16} \\ \hline \mathbf{39_{16}} \end{array}$$
右列：$3_{16} + 6_{16} = 3_{10} + 6_{10} = 9_{10} = 9_{16}$
左列：$2_{16} + 1_{16} = 2_{10} + 1_{10} = 3_{10} = 3_{16}$

(b)
$$\begin{array}{r} 58_{16} \\ +22_{16} \\ \hline \mathbf{7A_{16}} \end{array}$$
右列：$8_{16} + 2_{16} = 8_{10} + 2_{10} = 10_{10} = A_{16}$
左列：$5_{16} + 2_{16} = 5_{10} + 2_{10} = 7_{10} = 7_{16}$

(c) $2B_{16}$ 右列: $B_{16}+4_{16}=11_{10}+4_{10}=15_{10}=F_{16}$
$+84_{16}$ 左列: $2_{16}+8_{16}=2_{10}+8_{10}=10_{10}=A_{16}$
$\mathbf{AF_{16}}$

(d) DF_{16} 右列: $F_{16}+C_{16}=15_{10}+12_{10}=27_{10}$
$+AC_{16}$ $27_{10}-16_{10}=11_{10}=B_{16}$ 加1进位
$\mathbf{18B_{16}}$ 左列: $D_{16}+A_{16}+1_{16}=13_{10}+10_{10}+1_{10}=24_{10}$
$24_{10}-16_{10}=8_{10}=8_{16}$ 加1进位

相关问题: $4C_{16}$加上$3A_{16}$。

2.8.7 十六进制减法

如同前面介绍的,补码允许利用二进制数相加来进行减法的运算。由于十六进制数可以表示二进制数,所以也可以表示二进制数的补码。

有3种方法可以取得十六进制数的补码,方法1最常见也最容易使用,方法2和方法3是替代方法。

方法1 将十六进制数转换为二进制数。取二进制数的补码,把结果转换为十六进制数,如图2.3所示。

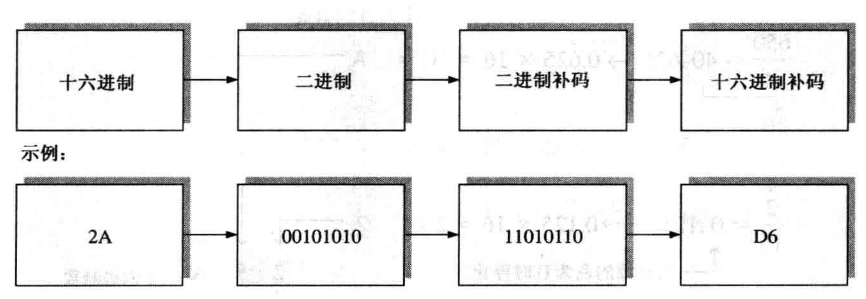


图2.3 得到十六进制数的补码的方法1

方法2 从最大十六进制数减去当前十六进制数,并加1,如图2.4所示。

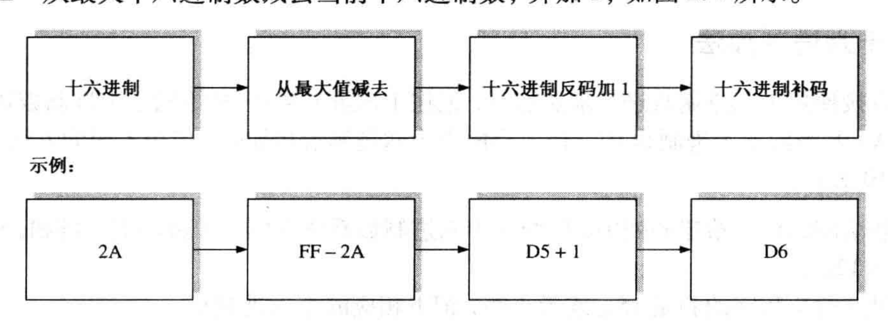


图2.4 得到十六进制数的补码的方法2

方法3 写出单个十六进制数的序列。在这一行的下面,以相反的顺序写出这个序列。每个十六进制数的反码就是该数字正下方的数字,结果数加1得到补码,如图2.5所示。

例2.30 将下面的十六进制数相减:

(a)$84_{16}-2A_{16}$ (b)$C3_{16}-0B_{16}$

解: (a)$2A_6=00101010$

$2A_{16}$的补码$=11010110=D6_{16}$(使用方法1)

$$\begin{array}{rl} 84_{16} & \\ +\ D6_{16} & \text{加} \\ \hline \not{1}5A_{16} & \text{丢弃进位，像补码加法一样} \end{array}$$

差是 **$5A_{16}$**。

(b) $0B_{16}=00001011$

$0B_{16}$ 的补码 $=11110101=F5_{16}$ (使用方法 1)

$$\begin{array}{rl} C3_{16} & \\ +\ F5_{16} & \text{加} \\ \hline \not{1}B8_{16} & \text{丢弃进位} \end{array}$$

差是 **$B8_{16}$**。

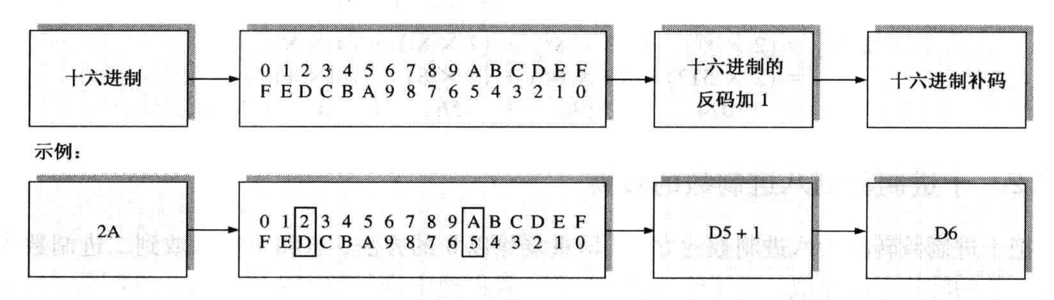


图 2.5 得到十六进制数的补码的方法 3

相关问题： 从 BCD_{16} 中减去 173_{16}。

2.8 节 温故而知新

1. 把下面的二进制数转换为十六进制数：

   (a) 10110011 (b) 110011101000

2. 把下面的十六进制数转换为二进制数：

   (a) 57_{16} (b) $3A5_{16}$ (c) $F80B_{16}$

3. 把 $9B\,30_{16}$ 转换为十进制数。

4. 把 573 转换为十六进制数。

5. 把下面的十六进制数相加：

   (a) $18_{16}+34_{16}$ (b) $3F_{16}+2A_{16}$

6. 把下面的十六进制数相减：

   (a) $75_{16}-21_{16}$ (b) $94_{16}-5C_{16}$

2.9 八进制数

八进制数字系统由 8 个数位组成，它们是

$$0, 1, 2, 3, 4, 5, 6, 7$$

要计大于 7 的数，则从另一列重新开始：

$$10, 11, 12, 13, 14, 15, 16, 17, 20, 21, \cdots$$

◇ 八进制数字系统的基是 8。

八进制计数和十进制计数比较相似，不同的是没有使用数字 8 和 9。为了区别八进制数和十进制数或者十六进制数，我们使用下标 8 来表示八进制数。例如，八进制 15_8 等于十进制数 13_{10} 和十六进制数 D。有时会看到八进制数后面跟随“o”或者“Q”。

2.9.1 八进制数到十进制数的转换

由于八进制数的基是 8，因此每一个相继的数位都是 8 的递增幂，从最右边 8^0 开始，计算八进制数的相应十进制数的数值，可以通过把每一个数字都乘以其相应的权，并把所得到的积加起来。下面展示了 2374_8 的转换过程。

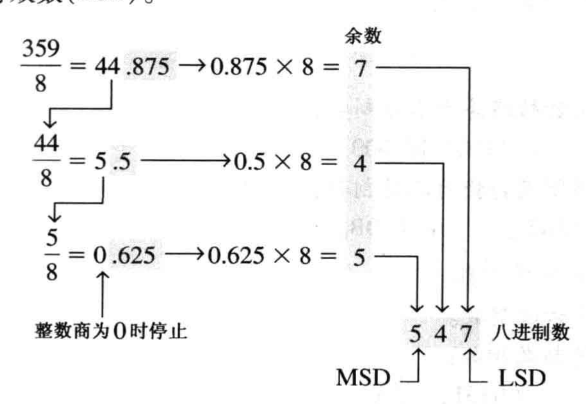
权：$8^3\ 8^2\ 8^1\ 8^0$

八进制数：2 3 7 4

$$
\begin{aligned}
2374_8 &= (2\times 8^3) + (3\times 8^2) + (7\times 8^1) + (4\times 8^0)\\
&= (2\times 512) + (3\times 64) + (7\times 8) + (4\times 1)\\
&= 1024 + 192 + 56 + 4 = 1276_{10}
\end{aligned}
$$

2.9.2 十进制数到八进制数的转换

把十进制数转换为八进制数的方法，即重复除以 8 的方法，这和十进制数到二进制数或十六进制数的转换方法相似。为了给出这个过程，我们把十进制数 359 转换为八进制数，以 8 为除数的每一次相继相除都会产生一个余数，而这个余数就会成为相应八进制数的数位。所产生的第一个余数是最低有效数(LSD)。



2.9.3 八进制数到二进制数的转换

◇ 八进制是表示二进制数的简洁方法，但是并没有十六进制数常用。

因为每一个八进制数位都可以由一个 3 位二进制数来表示，所以八进制数转换为二进制数很容易。每一个八进制数位由 3 位二进制数表示，如表 2.4 所示。

表 2.4 八进制/二进制转换

八进制	0	1	2	3	4	5	6	7
二进制	000	001	010	011	100	101	110	111

为了把八进制数转换为二进制数，只要用相应的 3 位二进制数替换每个八进制数位就可以了。这个过程如例 2.31 所示。

例2.31 把下面的八进制数转换为二进制数：

(a)13_8 (b)25_8 (c)140_8 (d)7526_8

解：(a) 1 3 → **001011**　(b) 2 5 → **010101**　(c) 1 4 0 → **001100000**　(d) 7 5 2 6 → **111101010110**

相关问题：将每一个二进制数转换为十进制数，并证明每一个值和相应八进制数的十进制值相等。

2.9.4 二进制数到八进制数的转换

二进制数到八进制数的转换是八进制数到二进制数的转换的逆过程。如下所述：从最右边的3位数一组开始，从右向左移动，把每3位数一组变换为相应的八进制数。如果最左边的一组没有可用的3位数，就加上一个或者两个0以形成3位一组。前面的0不影响二进制数的值。

例2.32 把下面每一个二进制数都转换为八进制数：

(a)110101 (b)101111001 (c)100110011010 (d)11010000100

解：(a) 110101 → 6 5 = $\mathbf{65_8}$　(b) 101111001 → 5 7 1 = $\mathbf{571_8}$　(c) 100110011010 → 4 6 3 2 = $\mathbf{4632_8}$　(d) 011010000100 → 3 2 0 4 = $\mathbf{3204_8}$

相关问题：把二进制数1010101000111110010转换为八进制数。

2.9节 温故而知新

1. 把下面的八进制数转换为十进制数：
   (a)73_8 (b)125_8
2. 把下面的十进制数转换为八进制数：
   (a)98_{10} (b)163_{10}
3. 把下面的八进制数转换为二进制数：
   (a)46_8 (b)723_8 (c)5624_8
4. 把下面的二进制数转换为八进制数：
   (a)110101111 (b)1001100010 (c)10111111001

2.10 二-十进制编码(BCD)

2.10.1 8421 BCD码

◇ 在BCD码中，每个十进制数都由4位二进制编码表示。

8421码是BCD(二-十进制编码)码的一种类型。二-十进制编码的意思是，每一个十进制数，从0~9，都由4位二进制编码表示。名称8421表明了4个位的二进制权(2^3, 2^2, 2^1, 2^0)。这种编码的主要优点是，8421编码数和我们熟悉的十进制数之间很容易转换。只要记住10个十进制数的二进制组合，如表2.5所示。8421码是主要的BCD码，所以当我们提及BCD码时，总是指8421码，除非有特殊的说明。

表 2.5 十进制/BCD 转换

十进制	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

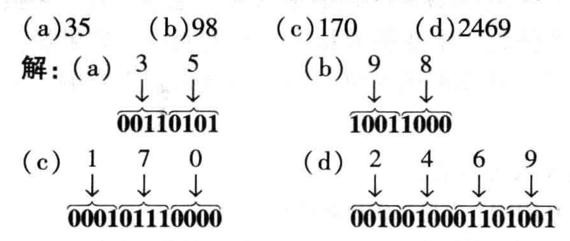
无效码 应当认识到，使用 4 个位可以表示 16 个数(从 0000 到 1111)，但是在 8421 码中，在这 16 个数中只使用了 10 个数。未使用的编码组合为 1010、1011、1100、1101、1110、1111，它们在 8421 BCD 码中是无效的。

为了以 BCD 码表示任意十进制数，只要将每个十进制数位用相应的 4 位编码替代就可以了，如例 2.33 所示。

例 2.33 把下面的十进制数转换为 BCD 码：

(a)35 (b)98 (c)170 (d)2469

解：



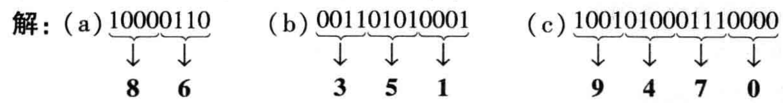
相关问题：把十进制数 9673 转换为 BCD 码。

同样，很容易根据一个 BCD 码确定一个十进制数。从最右边的一位开始，把 BCD 码分成 4 位一组。然后写出每个 4 位一组所表示的十进制数。例 2.34 给出了这个过程。

例 2.34 把下面的 BCD 码转换为十进制数：

(a)10000110 (b)001101010001 (c)1001010001110000

解：



相关问题：把 BCD 码 10000010001001110110 转换为十进制数。

应用 数字钟、数字温度计、数字仪表和一些其他使用七段显示器的设备是典型的使用 BCD 码显示十进制数的装置。进行运算时 BCD 码不如二进制数那么有效和直接，但是如果仅局限于所需要的处理(例如数字温度计)，就显得特别有用。



计算机小知识

BCD 码在计算机中有时用来进行算术运算。在计算机中表示 BCD 码时，通常要"压缩"使得 8 位数有两个 BCD 码。一般情况下，计算机就如同直接对二进制数进行相加一样。当 BCD 码相加或相减时，计算机程序员需要使用特殊的指令纠正运算结果。例如，在汇编语言里，程序中有 DAA(十进制加法调整)指令，用以自动纠正 BCD 码相加后的结果。

2.10.2 BCD 码加法

BCD 码是一种数字码，并且可以使用在算术运算中。加法是最重要的运算，因为其他 3 种运算(减法、乘法、除法)可以用加法来完成。下面介绍了如何将两个 BCD 码相加。

步骤 1：使用 2.4 节中二进制加法的规则，将两个 BCD 码相加。

步骤 2：如果 4 位和等于或者小于 9，这个和就是一个合法 BCD 码。

步骤 3：如果 4 位和大于 9，或者如果在 4 位一组之外产生了一个进位，那么这就是一个无

效结果。在 4 位和上加 6(0110) 以跳过 6 个无效状态，并将编码返回 8421 码。如果加上 6 时产生进位，就把这个进位加到下一个 4 位一组中。

例 2.35 说明了 BCD 码的加法，在这个例子中，4 位和等于或者小于 0，所以这些 4 位和都是有效的 BCD 码。例 2.36 说明了出现无效和(大于 9 或者有进位)的过程。

进行 BCD 码运算的一个替换方法是把它们转换成十进制数，进行加法运算，然后把运算结果转换回 BCD 码。

例 2.35 将下面的 BCD 码相加：

(a) 0011 + 0100　　(b) 00100011 + 00010101

(c) 10000110 + 00010011　　(d) 010001010000 + 010000010111

解： 同样也给出了十进制加法以做对比。

```
(a)   0011       3        (b)   0010  0011      23
    + 0100     + 4            + 0001  0101    + 15
      0111       7              0011  1000      38

(c)   1000  0110      86  (d)   0100  0101  0000      450
    + 0001  0011    + 13      + 0100  0001  0111    + 417
      1001  1001      99        1000  0110  0111      867
```

列的和都没有超出 9，所以这些结果都是有效的 BCD 码。

相关问题： 将 BCD 码相加：1001000001000011 + 0000100100100101 0。

例 2.36 将下面的 BCD 码相加：

(a) 1001 + 0100　　(b) 1001 + 1001

(c) 00010110 + 00010101　　(d) 01100111 + 01010011

解： 同样也给出了十进制加法以做对比。

```
(a)           1001                             9
            + 0100                            +4
              1101     无效的 BCD 码 (>9)      13
            + 0110     加 6
      0001    0011     有效的 BCD 码
       ↓       ↓
       1       3

(b)           1001                             9
            + 1001                            +9
         1    0010     由于进位无效            18
            + 0110     加 6
      0001    1000     有效的 BCD 码
       ↓       ↓
       1       8

(c)    0001    0110                                 16
     + 0001    0101                               + 15
       0010    1011     右边一组无效 (>9)            31
                        左边一组有效
             + 0110     加 6 变为有效码，
                        加进位，0001 到下一组
       0011    0001     有效的 BCD 码
        ↓       ↓
        3       1
```

```
(d)        0110     0111                         67
         + 0101     0011                       + 53
           1011     1010     两组都无效(>9)     120
         + 0110   + 0110     加6到两组
   0001    0010     0000     有效的BCD码
    ↓       ↓        ↓
    1       2        0
```

相关问题： 将 BCD 码相加：01001000 + 00110100。

2.10 节　温故而知新

1. 在下面的 BCD 码中，每个 1 所表示的权是多少？
   (a)0010　(b)1000　(c)0001　(d)0100
2. 把下面的十进制数转换为 BCD 码：
   (a)6　(b)15　(c)273　(d)849
3. 下面的每个 BCD 码所表示的十进制数是多少？
   (a)10001001　(b)001001111000　(c)000101010111
4. 在 BCD 码的加法中，无效的 4 位和发生在什么时候？

2.11　数字编码

2.11.1　格雷码

格雷(Gray)码是无权码，并不是算术编码；也就是没有赋予不同位的特定的权。格雷码的重要特征是，从一个码字到下一个接续码字仅有一位发生了变化。这个特征在许多应用程序中是非常重要的，例如对于轴位编码器，在两个相邻顺序数之间，错误敏感度随着位数改变数目的增加而增加。

◇ 格雷码的一位改变的特征减小了出错概率。

表 2.6 列出了十进制数 0～15 所对应的 4 位格雷码，表中给出了二进制数以做参照。和二进制数相似，格雷码可以拥有任意的位数。注意两个相邻的格雷码字之间的一位变化。例如，从十进制数 3 到十进制数 4，格雷码从 0010 变为 0110，而二进制编码从 0011 变为 0100，改变了 3 个位。格雷码中唯一的位改变是从右数的第 3 位，其他位保持不变。

表 2.6　4 位格雷码

十进制	二进制	格雷码	十进制	二进制	格雷码
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

二进制数到格雷码的转换　二进制数和格雷码之间的转换有时很有用。下面的规则解释了怎样把二进制数变换为格雷码。

1. 格雷码中的最高有效位(最左边)等同于二进制数中相应的最高有效位。
2. 从左到右，加上每一对相邻的二进制编码位，得到下一个格雷码位。舍去进位。

例如，二进制数 10110 到格雷码的转换如下：

```
1 - + → 0 - + → 1 - + → 1 - + → 0   二进制数
↓       ↓       ↓       ↓       ↓
1       1       1       0       1   格雷码
```

格雷码为 11101。

格雷码到二进制数的转换　为了把格雷码转为二进制数，可以使用相似的方法，但是有一些区别。可以应用下面的规则。

1. 二进制编码中的最高有效位(最左边)等于格雷码中相应的位。
2. 将所产生的每个二进制编码位加上下一相邻位置的格雷码位。舍去进位。

例如，格雷码 11011 到二进制的转换如下：

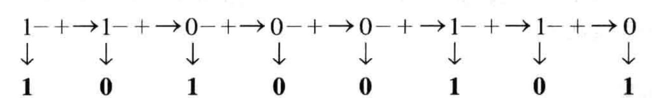
```
1        1        0        1        1    格雷码
↓  + ↗   ↓  + ↗   ↓  + ↗   ↓  + ↗   ↓
1 ↗      0 ↗      0 ↗      1 ↗      0    二进制数
```

二进制数是 10010。

例 2.37　(a)把二进制数 11000110 转换为格雷码。

(b)把格雷码 10101111 转换为二进制数。

解：(a)二进制数到格雷码：



(b)格雷码到二进制数：

```
1     0     1     0     1     1     1     1
↓ + ↗ ↓ + ↗ ↓ + ↗ ↓ + ↗ ↓ + ↗ ↓ + ↗ ↓ + ↗ ↓
1     1     0     0     1     0     1     0
```

相关问题：(a)把二进制数 101101 转换为格雷码。(b)把格雷码 100111 转换为二进制数。

2.11.2　典型应用

3 位轴位编码器的示意图如图 2.6 所示。通常，有三个同心圆环被分成 8 个扇区。扇区越多，位置就能表示得越准确，但是为了解释的方便，仅仅使用了 8 个扇区。每个圆环的每个扇区分为能够反射光束或不能反射光束两种，当圆环随着轴转动时，处于红外线(IR)发射器下部的扇区分别接收红外线发射器所产生的 3 条光束。当扇区反射光束时表示 1，当扇区不反射光束时表示 0。红外线探测器检测是否存在反射光束，然后产生相应的 3 位编码。红外线发射器/探测器处于固定位置，当轴逆时针 360°旋转时，8 个扇区在 3 条光束下移动。每一条光束被扇区的表面反射或吸收，产生表示轴位编码器位置的二进制数或格雷码。

在图 2.6(a)中，这些扇区直接以二进制格式排列，所以探测器输出从 000 到 001，再到 010，再到 011，等等。当光束处于反射扇区上部时，输出为 1；当光束处于非反射扇区上部时，输出为 0。如果从一个扇区到另一个扇区的转换瞬间，一条光束稍微先于其他光束被反射或吸

收，就会产生一个错误的输出。考虑这3条光束从111扇区变换到000扇区所发生的情况。如果最高位的光束略微超前，那么由于111或000被011瞬间替代，轴位将被错误地显示。在这个应用中，要使得红外发射器/探测器的安装精确对齐，实际上是不可能的，因此在扇区之间瞬间变换时，通常会发生错误。

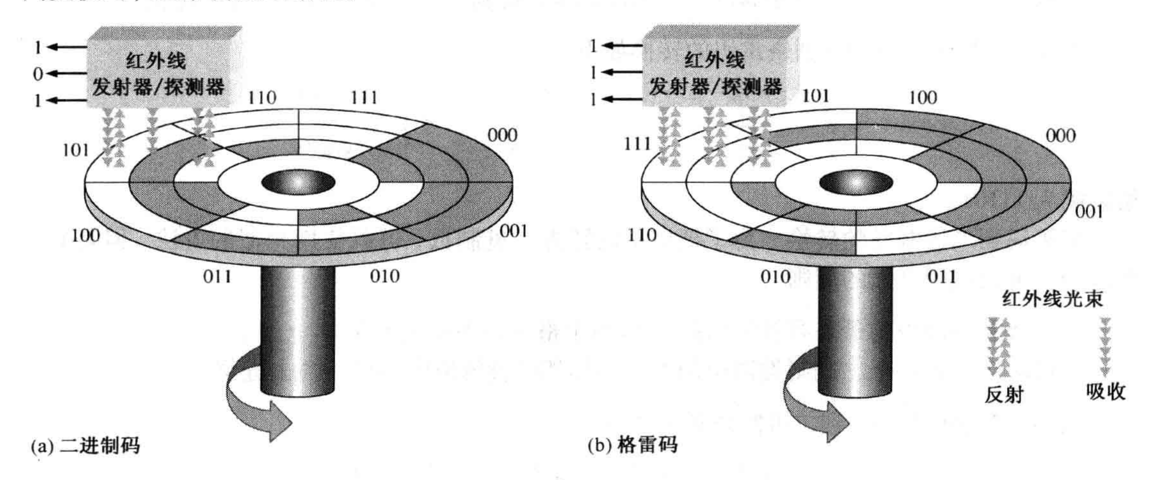


图2.6 一个简单的例子，解释在轴位编码器中格雷码是如何解决出错问题的。虽然大多数轴位编码器使用10位以上的编码以提高分辨率，但是使用3位就可以解释这种概念

格雷码用来消除二进制所固有的这类出错问题。如图2.6(b)所示，格雷码保证了在两个相邻的扇区中，只有一个位会发生改变。这就意味着即使光束没有精确对准，也绝对不会发生转移瞬间中的错误。例如，再一次考虑，当光束位于111扇区，准备进入下一个101扇区会发生什么情况。无论光束是否对准，在转移瞬间只有两个可能的输出111和101。在其他扇区的转移瞬间中，也会产生相同的结果。

2.11 节 温故而知新

1. 把下面的二进制数转换为格雷码：

   (a)1100　(b)1010　(c)11010

2. 把下面的格雷码转换为二进制数：

   (a)1000　(b)1010　(c)11101

2.12 错误检测码

2.12.1 错误检测的奇偶校验法

◇ 奇偶校验位给出一个数中1的个数是奇数还是偶数。

许多系统都使用一个奇偶校验位，作为位错误检测的手段。任意的多位数组都包含奇数个1或偶数个1。将一个奇偶校验位附加到多位数组中，使得这组数中1的个数总是偶数或者总是奇数。一个偶校验位使得1的总数为偶数，而奇校验位使得1的总数为奇数。

一个给定的系统运行于偶校验或者奇校验，而不是同时运行于两者。例如，如果某系统运行于偶校验，则对于所接收的每一个多位数组都做一个检查，以确保这个多位数组中1的总数是偶数。如果有奇数个1，就有一个错误发生了。

作为对奇偶校验位怎样附加到编码中的一个说明，表 2.7 列出了每个 BCD 码的偶校验和奇校验的奇偶校验位。每个 BCD 码的校验位处于 *P* 列上。

表 2.7　带奇偶校验位的 BCD 码

偶校验		奇校验	
校验位 *P*	BCD	校验位 *P*	BCD
0	0000	1	0000
1	0001	0	0001
1	0010	0	0010
0	0011	1	0011
1	0100	0	0100
0	0101	1	0101
0	0110	1	0110
1	0111	0	0111
1	1000	0	1000
0	1001	1	1001

奇偶校验位可以附加到码的开头或者结尾，这取决于系统的设计。注意，1 的总数，包括奇偶校位上的 1，对于偶校验总是偶数，对于奇校验总是奇数。

检测一个错误　奇偶校验位提供了单个位错误的检测（或者任何奇数个错误，这种可能性较小），但是不能检测一组数中的两个错误。例如，假设我们希望传送 BCD 码 0101。（奇校验可以应用于任何位数的码；使用 4 位作为说明。）所传送的总的码，包括偶校验位如下所示：

偶校验位

00101

BCD 码

现在，让我们假设从左边数第 3 位发生了错误（1 变为 0），如下所示：

偶校验位

00001

位错误

当该码被接收时，奇偶校验检测电路检测出只有一个 1（奇数），但是应当有偶数个 1。因为在码被接收时，偶数个 1 没有出现在码中，所以就指出了一个错误。

奇校验位提供了相似的方式，用以检测给定位数的数组的单个错误。

例 2.38　给下面的数组加上适当的偶校验位：

(a) 1010　(b) 111000　(c) 101101

(d) 1000111001001　(e) 101101011111

解：根据需要使奇偶校验位为 1 或者 0，使得 1 的总数为偶数。奇偶校验位将是最左边的一位。

(a) **0**1010　(b) **1**111000　(c) **0**101101

(d) **0**100011100101　(e) **1**101101011111

相关问题：为字母 K 的 7 位 ASCII 码加上一个偶校验位。

例 2.39　一个奇校验系统接收到下面的数组：10110、11010、110011、110101110100 和 1100010101010。如果有错误，请确定哪一组是错误的。

解：由于需要奇校验，所以任何一个拥有偶数个1的数组就是错误的。下面的数组有错误：**110011** 和 **1100010101010**。

相关问题：一个奇校验系统接收到下列ASCII字符：00110111。这个数组正确吗？

2.12.2 循环冗余校验码(CRC)

在通信链路的数字数据的传送过程中，循环冗余校验码(CRC)广泛用于检测一位或两位传送错误。通信链路存在于两台通过网络相连的计算机之间，或者是一个数字存储装置(例如CD、DVD或硬盘)和一台PC之间。如果设计合理，循环冗余校验码还可以检测一组比特数中的多个连续的错误(区间误差)。在循环冗余校验码中，检测位的数目有时称为校验和，它被加到传送的数据位中(加在最后)。接收器使用循环冗余校验码来检测接收到的数据的错误。循环冗余校验码并不能检测出所有可能出现的错误，但是它比简单的奇偶校验要有效得多。

循环冗余校验码在数学上通常描述为两个多项式相除，并产生一个余数。多项式就是一个数学表达式，这个表达式具有多个正指数项的和。如果各项系数仅为1和0时，就称为单变量多项式。单变量多项式的一个例子是$1x^3+0x^2+1x^1+1x^0$，或简写为$x^3+x^1+x^0$，这完全可以用4位二进制数1011来表示。大多数循环冗余校验码使用16位或较大的多项式，但是为了解释简单，这里使用4位。

表2.8 模-2运算

输入位	输出位
0 0	0
0 1	1
1 0	1
1 1	0

模-2运算 可以简单地认为，循环冗余校验码就是基于两个二进制数的相除，除法就是一系列的减法和移位。进行减法操作时，使用一个称为模-2加法的方法。模-2加法(或减法)和不考虑进位的二进制加法相同，如表2.8的真值表所示。如同第3章将要学习的那样，真值表广泛地用于描述逻辑电路的运算。两位数位在真值表中有四种可能的组合。这个特定的表描述的模-2运算，也称为异或运算，可以由第3章将要介绍的逻辑门来实现。模-2运算的一个简单规则就是如果两个输入不同，输出就是1，否则输出为0。

循环冗余校验码的处理步骤 如下所示：

1. 选择一个固定的生成码，它可以比校验的数据位的位数少。这个生成码要事先让发送器和接收器都能识别，并且对于发送器和接收器必须都是相同的。
2. 附加上若干个0，0的个数和加到数据位的生成码的0的个数相同。
3. 使用模-2方法，把数据位和附加的生成位合在一起除以生成码。
4. 如果余数为0，那么数据位和附加的位原样发送。
5. 如果余数不为0，那么为了使发送数据前余数为0，附加的位改为余数位。
6. 在接收端，接收器使用和发送器相同的生成数位除以接收到的附加的数据位。
7. 如果余数为0，就没有检测到错误(多个错误的可能性很小，使用时不考虑这种情况)。如果余数不为0，就检测到在传送过程中有一个错误，接收器会发出一个重传请求。

图2.7为循环冗余校验码的处理步骤的示意图。

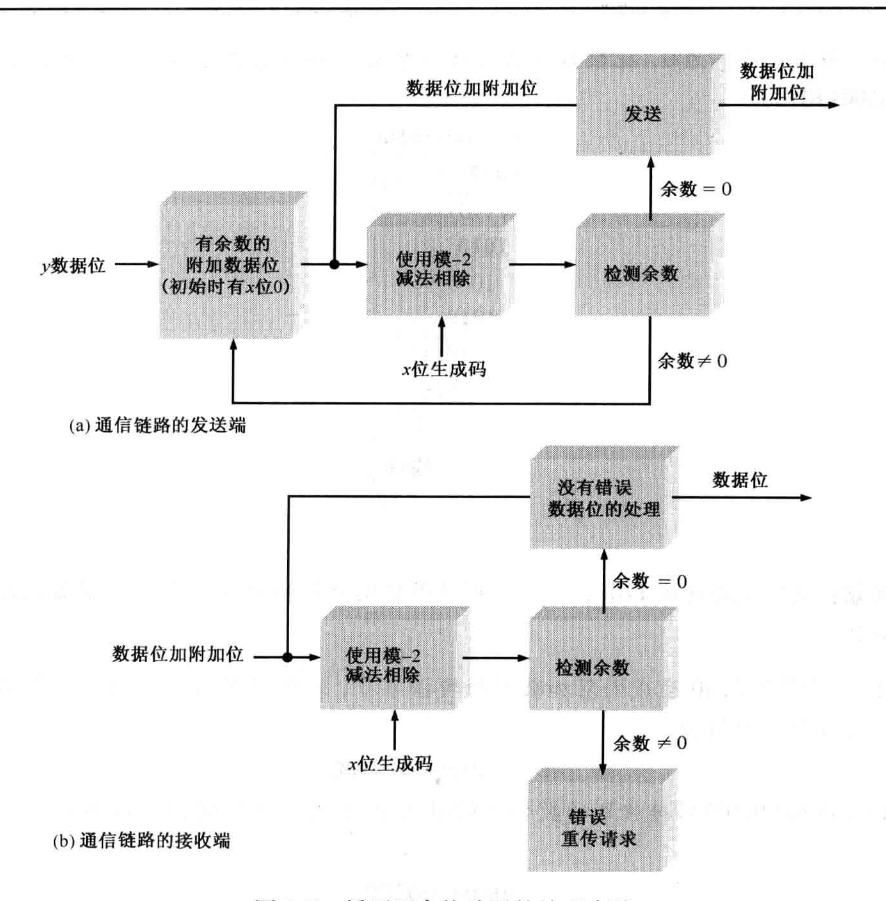


图2.7 循环冗余校验码的处理步骤

例2.40 为下面的数据(D)字节和生成码(G)确定循环冗余校验码。检查余数是否为0。

D: 11010011

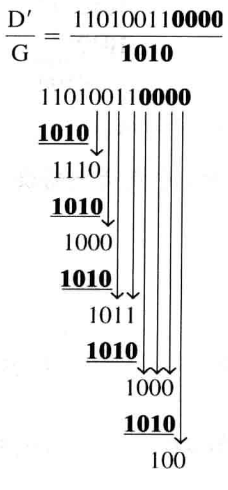
G: 1010

解: 因为生成码有4位，所以数据字节添加4个0，附加的数据(D′)为

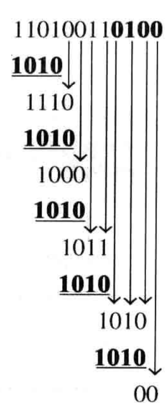
$$D' = 11010011\mathbf{0000}$$

使用模-2运算把附加后得到的数据除以生成码，直到所有位都运算过。

$$\frac{D'}{G} = \frac{11010011\mathbf{0000}}{\mathbf{1010}}$$



余数 =0100。由于余数不为 0，把数据位加上 4 位余数。然后除以生成码。传送的循环冗余校验码为 **11010011**0100。



余数 =0。

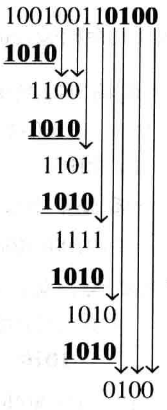
相关问题： 把生成码改为 1100，然后检测当循环冗余校验码作用于数据字节(11010011)时余数是否为 0。

例 2.41 对于例 2.40 生成的附加数据的数据字节，在传送过程中左边开始第二位发生了一个错误。接收到的数据为

D′ = 10010011**0100**

使用相同的生成码(1010)对接收到的数据进行错误检测(使用循环冗余校验码)。

解：



余数 =0100。由于余数不为 0，表明有一个错误。

相关问题： 假设在数据字节 10011011 中有两个错误。使用同样接收到的数据和生成码，应用循环冗余校验码来检测错误。

2.12 节 温故而知新

1. 下面哪一个奇校验码有错误？

(a)1011　　(b)1110　　(c)0101　　(d)1000

2. 下面哪一个偶校验码有错误？

(a)11000110　　(b)00101000　　(c)10101010　　(d)11111011

3. 在下面每个码的最后添加一个偶校验位。
   (a)1010100 (b)0100000 (c)1110111(d)1000110
4. CRC 表示什么?
5. 应用模-2运算确定下面的结果:
   (a)1+1 (b)1-1 (c)1-0 (d)0+1

关键词

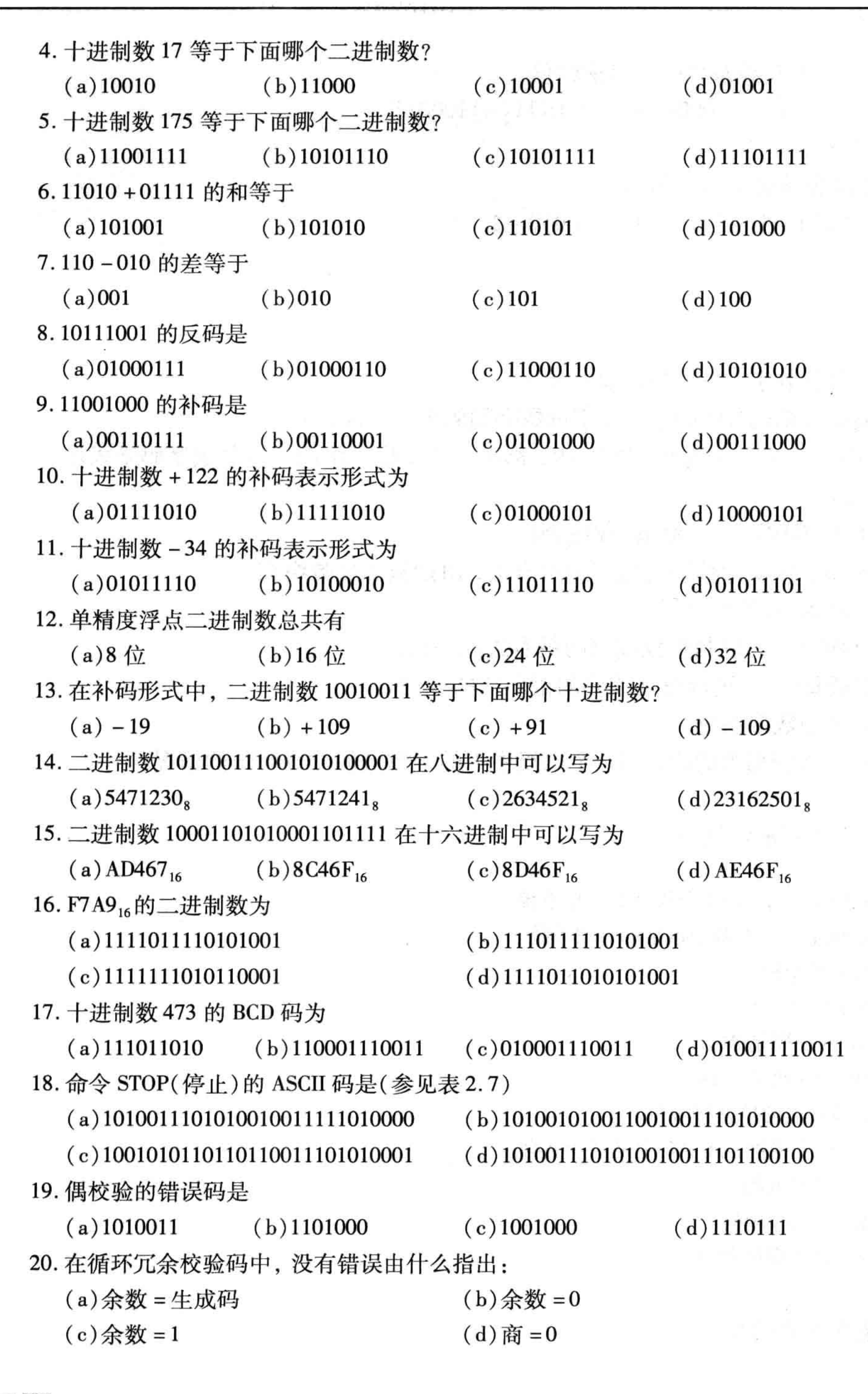
字母数字的 包括数字、字母和其他的文字。
美国信息交换标准编码(ASCII) 在字母数字编码中广泛应用。
二-十进制编码 每个十进制数，即0~9，都由一个4位一组的二进制数字码来表示。
字节 8位一组。
循环冗余校验码(CRC) 一种错误检测码。
浮点数 一种基于科学计数法的数字表示方法，由指数和尾数组成。
十六进制数 基为16的数字系统。
最低有效位(LSB) 二进制整数或码的最右边的一位。
最高有效位(MSB) 二进制整数或码的最左边的一位。
八进制 基为8的数字系统。
奇偶校验 与二进制相关的码，即一个码组中1的个数是偶数或奇数的情况。

判断题 (答案在本章的结尾。)

1. 十进制数字系统是一个有10个数字的有权系统。
2. 二进制数字系统是一个有两个数字的有权系统。
3. LSB表示最低的单个位。
4. 在二进制数中：1+1=2。
5. 二进制数1010的反码是0101。
6. 二进制数0001的补码是1110。
7. 在符号数中，最右边的位是符号位。
8. 十六进制数有16个字符，其中6个是字母字符。
9. BCD表示二-十进制编码。
10. CRC表示循环冗余校验码。
11. 11和10的模-2运算的和是100。

自测题 (答案在本章的结尾。)

1. $2\times10^1+8\times10^0$等于
   (a)10 (b)280 (c)2.8 (d)28
2. 二进制数1101等于下面哪个十进制数?
   (a)13 (b)49 (c)11 (d)3
3. 二进制数11011101等于下面哪个十进制数?
   (a)121 (b)221 (c)441 (d)256



4. 十进制数 17 等于下面哪个二进制数?

(a)10010 (b)11000 (c)10001 (d)01001

5. 十进制数 175 等于下面哪个二进制数?

(a)11001111 (b)10101110 (c)10101111 (d)11101111

6. 11010 + 01111 的和等于

(a)101001 (b)101010 (c)110101 (d)101000

7. 110 - 010 的差等于

(a)001 (b)010 (c)101 (d)100

8. 10111001 的反码是

(a)01000111 (b)01000110 (c)11000110 (d)10101010

9. 11001000 的补码是

(a)00110111 (b)00110001 (c)01001000 (d)00111000

10. 十进制数 +122 的补码表示形式为

(a)01111010 (b)11111010 (c)01000101 (d)10000101

11. 十进制数 -34 的补码表示形式为

(a)01011110 (b)10100010 (c)11011110 (d)01011101

12. 单精度浮点二进制数总共有

(a)8 位 (b)16 位 (c)24 位 (d)32 位

13. 在补码形式中,二进制数 10010011 等于下面哪个十进制数?

(a) -19 (b) +109 (c) +91 (d) -109

14. 二进制数 101100111001010100001 在八进制中可以写为

(a)5471230_8 (b)5471241_8 (c)2634521_8 (d)23162501_8

15. 二进制数 10001101010001101111 在十六进制中可以写为

(a)$AD467_{16}$ (b)$8C46F_{16}$ (c)$8D46F_{16}$ (d)$AE46F_{16}$

16. $F7A9_{16}$的二进制数为

(a)1111011110101001 (b)1110111110101001

(c)1111111010110001 (d)1111011010101001

17. 十进制数 473 的 BCD 码为

(a)111011010 (b)110001110011 (c)010001110011 (d)010011110011

18. 命令 STOP(停止)的 ASCII 码是(参见表 2.7)

(a)10100111010100100111111010000 (b)10100101001100100111010100000

(c)10010101101101100111010100001 (d)1010011101010010011101100100100

19. 偶校验的错误码是

(a)1010011 (b)1101000 (c)1001000 (d)1110111

20. 在循环冗余校验码中,没有错误由什么指出:

(a)余数 = 生成码 (b)余数 = 0

(c)余数 = 1 (d)商 = 0

习题 (奇数题的答案在本书的结尾。)

2.1 节 十进制数

1. 下面的每一个十进制数中,数字 6 的权各是多少?

(a)1386 (b)54692 (c)671920

2. 把下面的每一个十进制数表示为 10 的幂数：

(a)10　(b)100　(c)10000　(d)1000000

3. 求出下面十进制数中每一位数的值：

(a)471　(b)9356　(c)125000

4. 使用 4 个十进制数位最大能数到几？

2.2 节　二进制数

5. 把下面每一个二进制数都转换为十进制数：

(a)11　(b)100　(c)111　(d)1000

(e)1001　(f)1100　(g)1011　(h)1111

6. 把下面每一个二进制数都转换为十进制数：

(a)1110　(b)1010　(c)11100　(d)10000

(e)10101　(f)11101　(g)10111　(h)11111

7. 把下面每一个二进制数都转换为十进制数：

(a)110011.11　(b)101010.01　(c)1000001.111　(d)1111000.101

(e)1011100.10101　(f)1110001.0001　(g)1011010.1010　(h)1111111.11111

8. 求下面所列出位数的二进制数(位)所能表示的最大十进制数分别是多少？

(a)2　(b)3　(c)4　(d)5　(e)6

(f)7　(g)8　(h)9　(i)10　(g)11

9. 表示下面的十进制数分别需要多少位？

(a)17　(b)35　(c)49　(d)68

(e)81　(f)114　(g)132　(h)205

10. 为下面每一个十进制序列分别生成二进制序列：

(a)0 ~ 7　(b)8`~ 15　(c)16 ~ 31　(d)32 ~ 63　(e)64 ~ 75

2.3 节　十进制数到二进制数的转换

11. 使用权值和的方法把每一个十进制数都转换为二进制数：

(a)10　(b)17　(c)24　(d)48

(e)61　(f)93　(g)125　(h)186

12. 使用权值的和方法把每一个十进制数都转换为 *M* 进制数：

(a)0.32　(b)0.246　(c)0.0981

13. 使用重复除以 2 的方法将每一个十进制数都转换为二进制数：

(a)15　(b)21　(c)28　(d)34

(e)40　(f)59　(g)65　(h)73

14. 使用重复乘 2 的方法把每一个十进制小数都转换为二进制数：

(a)0.98　(b)0.347　(c)0.9028

2.4 节　二进制算术

15. 执行下面的二进制数的加法：

(a)11 + 01　(b)10 + 10　(c)101 + 11　(d)111 + 110

(e)1001 + 101　(f)1101 + 1011

16. 对下面的二进制数使用直接减法：

(a)11 - 1　(b)101 - 100　(c)110 - 101

(d)1110 - 11　(e)1100 - 1001　(f)11010 - 10111

17. 执行下面的二进制数的乘法：

(a)11 × 11　(b)100 × 10　(c)111 × 101

(d)1001×110 (e)1101×1101 (f)1110×1101

18. 执行下面的二进制数的除法:

(a)100÷10 (b)1001÷11 (c)1100÷100

2.5 节 二进制数的反码和补码

19. 在反码中两种0的表示方法是什么?
20. 在补码中的表示方法是什么?
21. 确定下面每一个二进制数的反码:

(a)101 (b)110 (c)1010
(d)11010111 (e)1110101 (f)00001

22. 使用两种方法,确定下面每一个二进制数的补码:

(a)10 (b)111 (c)1001 (d)1101
(e)11100 (f)10011 (g)10110000 (h)00111101

2.6 节 带符号数

23. 把下面的每一个十进制数都表示为8位带符号二进制数:

(a) +29 (b) -85 (c) +100 (d) -123

24. 把下面的每一个十进制数都以反码形式表示为一个8位数:

(a) -34 (b) +57 (c) -99 (d) +115

25. 把下面的每一个十进制数都以补码形式表示为一个8位数:

(a) +12 (b) -68 (c) +101 (d) -125

26. 以符号数值形式确定每个带符号二进制数的十进制数值:

(a)10011001 (b)01110100 (c)10111111

27. 以反码形式确定每个带符号二进制数的十进制数值:

(a)10011001 (b)01110100 (c)10111111

28. 以补码形式确定每个带符号二进制数的十进制数值:

(a)10011001 (b)01110100 (c)10111111

29. 以单精度浮点数表示下面的每一个带符号二进制数:

(a)0111110000101011 (b)100110000011000

30. 确定下面单精度浮点数的数值:

(a)1 10000001 01001001110001000000000
(b)0 11001100 10000111110100100000000

2.7 节 带符号数的算术运算

31. 把下面的每对十进制数都转换为二进制数,并使用补码形式进行相加运算:

(a)33, 15 (b)56, -27 (c) -46, 25 (d) -110, -84

32. 以补码形式执行下面的每一个加法:

(a)00010110 +00110011 (b)01110000 +10101111

33. 以补码形式执行下面的每一个加法:

(a)100011000 +00111001 (b)11011001 +11100111

34. 以补码形式执行下面的每一个减法:

(a)00110011 -00010000 (b)01100101 -11101000

35. 以补码形式用11110001乘以01101010。
36. 以补码形式用01000100除以00011001。

2.8 节 十六进制数

37. 把每一个十六进制数都转换为二进制数:

(a) 38_{16}　(b) 59_{16}　(c) $A14_{16}$　(d) $5C8_{16}$
(e) 4100_{16}　(f) $FB17_{16}$　(g) $8A9D_{16}$

38. 把每一个二进制数都转换为十六进制数：
(a) 1110　(b) 10　(c) 10111
(d) 10100110　(e) 1111110000　(f) 100110000010

39. 把每一个十六进制数都转换为十进制数：
(a) 23_{16}　(b) 92_{16}　(c) $1A_{16}$　(d) $8D_{16}$
(e) $F3_{16}$　(f) EB_{16}　(g) $5C2_{16}$　(h) 700_{16}

40. 把每一个十进制数都转换为十六进制数：
(a) 8　(b) 14　(c) 33　(d) 52
(e) 284　(f) 2890　(g) 4019　(h) 6500

41. 执行下面的加法：
(a) $37_{16}+29_{16}$　(b) $A0_{16}+6B_{16}$　(c) $FF_{16}+BB_{16}$

42. 执行下面的减法：
(a) $51_{16}-40_{16}$　(b) $C8_{16}-3A_{16}$　(c) $FD_{16}-88_{16}$

2.9 节　八进制数

43. 把下面每一个八进制数都转换为十进制数：
(a) 12_8　(b) 27_8　(c) 56_8　(d) 64_8　(e) 103_8
(f) 557_8　(g) 163_8　(h) 1024_8　(i) 7765_8

44. 使用重复除以 8 的方法，把下面每一个十进制数都转换为八进制数：
(a) 15　(b) 27　(c) 46　(d) 70
(e) 100　(f) 142　(g) 219　(h) 435

45. 把下面每一个八进制数都转换为二进制数：
(a) 13_8　(b) 57_8　(c) 101_8　(d) 321_8　(e) 540_8
(f) 4653_8　(g) 13271_8　(h) 45600_8　(i) 100213_8

46. 把下面每一个二进制数都转换为八进制数：
(a) 111　(b) 10　(c) 110111　(d) 101010
(e) 1100　(f) 1011110　(g) 101100011001　(h) 10110000011
(i) 11111101111000

2.10 节　二-十进制编码(BCD)

47. 把下面每一个十进制数都转换为 8421 BCD 码：
(a) 10　(b) 13　(c) 18　(d) 21　(e) 25　(f) 36
(g) 44　(h) 57　(i) 69　(j) 98　(k) 125　(l) 156

48. 把习题 47 中的每个十进制数都转换为直接的二进制数，比较两者所需要的位数。

49. 把下面的十进制数转换为 BCD 码：
(a) 104　(b) 128　(c) 132　(d) 150　(e) 186
(f) 210　(g) 359　(h) 547　(i) 1051

50. 把下面的每一个 BCD 码都转换为十进制数：
(a) 0001　(b) 0110　(c) 1001　(d) 00011000　(e) 00011001
(f) 00110010　(g) 01000101　(h) 10011000　(i) 100001110000

51. 把下面的每一个 BCD 码都转换为十进制数：
(a) 10000000　(b) 001000110111
(c) 001101000110　(d) 010000100001

(e)011101010100 (f)100000000000

(g)100101111000 (h)0001011010000011

(i)1001000000011000 (j)0110011001100111

52. 相加下面的 BCD 码:

(a)0010 +0001 (b)0101 +0011

(c)0111 +0010 (d)1000 +0001

(e)00011000 +00010001 (f)01100100 +00110011

(g)01000000 +01000111 (h)10000101 +00010011

53. 相加下面的 BCD 码:

(a)1000 +0110 (b)0111 +0101

(c)1001 +1000 (d)1001 +0111

(e)00100101 +00100111 (f)01010001 +01011000

(g)10011000 +10010111 (h)010101100001 +011100001000

54. 把下面的每对十进制数都转换为 BCD 码,并且执行加法运算:

(a)4 +3 (b)5 +2 (c)6 +4 (d)17 +12

(e)28 +23 (f)65 +58 (g)113 +101 (h)295 +157

2.11 节 数字编码

55. 在某个应用中,一个 4 位二进制序列从 1111 到 0000 周期性地循环。有 4 个数位发生变化,由于电路延迟,这些变化可能不会在同一瞬间出现。例如,如果最低有效位(LSB)首先变化,该数将在 1111 到 0000 的转换期间显示为 1110,并且可能被系统误认。解释格雷码是怎样避免这个问题的。

56. 把每一个二进制数都转换为格雷码:

(a)11011 (b)1001010 (c)1111011101110

2.12 节 错误检测码

57. 确定下面哪一个偶校验码有错:

(a)100110010 (b)011101010 (c)10111111010001010

58. 确定下面哪一个奇校验码有错:

(a)11110110 (b)00110001 (c)01010101010101010

59. 给下面的每一个数据字节添加合适的偶校验位:

(a)10100100 (b)00001001 (c)11111110

60. 使用模-2 方法进行如下运算:

(a)1100 +1011 (b)1111 +0100 (c)10011001 +100011100

61. 在习题 60 中把运算结果和原来的两个数相加得到另一个数,以此来验证模-2 减法和模-2 加法是相同的。这说明尽管两个数不同,但结果是一样的。

62. 使用生成码 1010 对数据位 10110010 进行循环冗余校验码编码,以产生传送的循环冗余校验码。

63. 假设习题 62 产生的码在传送过程中引起最高有效位有一个错误。使用循环冗余校验码来检测此错误。

答案

温故而知新

2.1 节 十进制数

1. (a)1370:10 (b)6725:100 (c)7051:1000 (d)58.72:0.1

2. (a)51 = (5 × 10) + (1 × 1)　(b)137 = (1 × 100) + (3 × 10) + (7 × 1)
(c)1492 = (1 × 1000) + (4 × 100) + (9 × 10) + (2 × 1)
(d)106.58 = (1 × 100) + (0 × 10) + (6 × 1) + (5 × 0.1) + (8 × 0.01)

2.2 节　二进制数

1. $2^8 - 1 = 255$
2. 权是 16。
3. 10111101.011 = 189.375

2.3 节　十进制数到二进制数的转换

1. (a)23 = 10111　(b)57 = 111001　(c)45.5 = 101101.1
2. (a)14 = 1110　(b)21 = 10101　(c)0.375 = 0.011

2.4 节　二进制算术

1. (a)1101 + 1010 = 10111　(b)10111 + 01101 = 100100
2. (a)1101 − 0100 = 1001　(b)1001 − 0111 = 0010
3. (a)110 × 111 = 101010　(b)1100 ÷ 011 = 100

2.5 节　二进制数的反码和补码

1. (a)00011010 的反码 = 11100101　(b)11110111 的反码 = 00001000
(c)10001101 的反码 = 01110010
2. (a)00010110 的补码 = 11101010　(b)11111100 的补码 = 00000100
(c)10010001 的补码 = 01101111

2.6 节　带符号数

1. 符号 − 大小：+9 = 00001001　2. 反码：−33 = 11011110
3. 补码：−46 = 11010010　4. 符号位，指数，尾数

2.7 节　带符号数的算术运算

1. 加法的情况：正数大，结果为正；负数大，结果为负。
2. 00100001 + 10111100 = 11011101
3. 01110111 − 00110010 = 01000101
4. 乘法结果的符号为正。
5. 00000101 × 01111111 = 01001111011
6. 商的符号为负。
7. 00110000 ÷ 00001100 = 00000100

2.8 节　十六进制数

1. (a)10110011 = $B3_{16}$　(b)110011101000 = $CE8_{16}$
2. (a)57_{16} = 01010111　(b)$3A5_{16}$ = 001110100101
(c)$F8OB_{16}$ = 1111100000001011
3. $9B30_{16} = 39.728_{10}$
4. $573_{10} = 23D_{16}$
5. (a)$18_{16} + 34_{16} = 4C_{16}$　(b)$3F_{16} + 2A_{16} = 69_{16}$
6. (a)$75_{16} - 21_{16} = 54_{16}$　(b)$94_{16} - 5C_{16} = 38_{16}$

2.9 节　八进制数

1. (a)$73_8 = 59_{10}$　(b)$125_8 = 85_{10}$

2. (a)$98_{10}=142_8$ (b)$163_{10}=243_8$

3. (a)$46_8=100110$ (b)$723_8=111010011$ (c)$5624_8=101110010100$

4. (a)$110101111=657_8$ (b)$1001100010=1142_8$ (c)$10111111001=2771_8$

2.10 节 二-十进制编码(BCD)

1. (a)0010:2 (b)1000:8 (c)0001:1 (d)0100:4

2. (a)$6_{10}=0110$ (b)$15_{10}=00010101$
 (c)$273_{10}=001001110011$ (d)$849_{10}=100001001001$

3. (a)$10001001=89_{10}$ (b)$001001111000=278_{10}$ (c)$000101010111=157_{10}$

4. 当4位和大于9_{10}时无效。

2.11 节 数字编码

1. (a)$1100_2=1010$(格雷码) (b)$1010_2=1111$(格雷码) (c)$11010_2=10111$(格雷码)

2. (a)1000(格雷码)$=1111_2$ (b)1010(格雷码)$=1100_2$ (c)11101(格雷码)$=10110_2$

2.12 节 错误检测码

1. (c)0101 有一个错误。 2. (d)11111011 有一个错误。

3. (a)10101001 (b)01000001 (c)11101110 (d)10001101

4. 循环冗余校验码

5. (a)0 (b)0 (c)1 (d)1

例题的相关问题

2.1 9 的值是 900, 3 的值是 30, 9 的值是 9。

2.2 6 的值是 60, 7 的值是 7, 9 的值是 0.9, 2 的值是 0.02, 4 的值是 0.004。

2.3 $10010001=128+16+1=145$ 2.4 $10.111=2+0.5+0.25+0.125=2.875$

2.5 $125=64+32+16+8+4+1=1111101$ 2.6 $39=100111$

2.7 $1111+1100=11011$ 2.8 $111-100=011$ 2.9 $110-101=001$

2.10 $1101\times1010=10000010$ 2.11 $1100\div100=11$

2.12 00110101 2.13 01000000 2.14 见表 2.9。

表 2.9

	符号-值	反 码	补 码
+19	00010011	00010011	00010011
−19	10010011	11101100	11101101

2.15 $01110111=+119_{10}$ 2.16 $11101011=-20_{10}$ 2.17 $11010111=-41_{10}$

2.18 11000010001010011000000000 2.19 01010101 2.20 00010001

2.21 1001000110 2.22 $(83)(-59)=-4897$(补码 10110011011111)

2.23 $100\div25=4(0100)$ 2.24 $4F79C_{16}$ 2.25 0110101111010011_2

2.26 $6BD_{16}=011010111101=2^{10}+2^9+2^7+2^5+2^4+2^3+2^2+2^0$
$=1024+512+128+32+16+8+4+1=1725_{10}$

2.27 $60A_{16}=(6\times256)+(0\times16)+(10\times1)=1546_{10}$ 2.28 $2591_{10}=A1F_{16}$

2.29 $4C_{16}+3A_{16}=86_{16}$ 2.30 $BCD_{16}-173_{16}=A5A_{16}$

2.31 (a)$001011_2=11_{10}=13_8$ (b)$010101_2=21_{10}=25_8$
(c)$001100000_2=96_{10}=140_8$ (d)$111101010110_2=3926_{10}=7526_8$

2.32 1250762_8 2.33 1001011001110011 2.34 $82\,276_{10}$

2.35 1001100101101000 2.36 10000010 2.37 (a)111011(格雷码) (b)111010_2

2.38 01001011 2.39 是 2.40 余数的结果为0。 2.41 表示错误。

判断题

1.T 2.T 3.F 4.F 5.T 6.F 7.F 8.T 9.T 10.T 11.F

自测题

1.(d) 2.(a) 3.(b) 4.(c) 5.(c) 6.(a) 7.(d) 8.(b)

9.(d) 10.(a) 11.(c) 12.(d) 13.(d) 14.(b) 15.(c) 16.(a)

17.(c) 18.(a) 19.(b) 20.(b)

第3章 逻 辑 门

章节提纲

固定功能逻辑器件(CMOS 和双极型系列)

74XX00 74XX02 74XX04 74XX08 74XX10 74XX11
74XX20 74XX21 74XX27 74XX30 74XX32 74XX86 74XX266

3.1 反相器

反相器的标准逻辑符号如图3.1所示，(a)图给出了特殊形状符号，(b)图给出了矩形轮廓符号。在这本书中，一般使用特殊形状符号；但是，矩形轮廓符号常常出现在许多工业出版物中，所以也应该熟悉它们。(逻辑门符号依据 ANSI/IEEE 标准 91-1984。)

(a)具有否定指示的特殊形状符号

(b)具有极性指示的矩形轮廓符号

图3.1 反相器的标准逻辑符号(ANSI/IEEE标准91-1984)

3.1.1 否定和极性指示

否定指示是一个小圆圈(○)，当其出现在任何逻辑元件的输入或者输出位置时，为反相或者为反码，如图3.1(a)的反相器所示。一般情况下，输入位于逻辑符号的左侧而输出位于右侧。当出现在输入位置时，小圆圈就表示0电平有效或者是确定的输入状态，而这个输入称为低电平有效输入。当出现在输出位置时，该小圆圈就指明0有效或者是确定的输出状态，而这个输出称为低电平有效输出。当输入或者输出没有小圆圈时，就表示1是有效的或者是确定的状态，而这个输入或输出称为高电平有效。

极性或者电平指示是一个“三角形”(◺)，当其出现在任何逻辑元件的输入或者输出位置时，表示反相，如图3.1(b)所示。当其出现在输入位置时，就表示低电平是有效的或者是确定的输入状态。当其出现在输出位置时，就表示低电平是有效的或者是确定的输出状态。

两种指示(小圆圈或三角形)都可以用在特殊形状符号和矩形轮廓符号中。图3.1(a)给出的是本书后面主要使用的反相器。注意反相或极性指示的不同放置并不意味着反相器运算方式的改变。

3.1.2 反相器真值表

当反相器的输入是高电平时，它的输出就是低电平。当反相器的输入是低电平时，它的输出就是高电平。这种运算总结于表3.1中，表中以电平和对应的位值给出了每个可能的输入和与之对应的输出。这样的表称为真值表。

表3.1 反相器真值表

输入	输出
低(0)	高(1)
高(1)	低(0)

3.1.3 反相器运算

图3.2给出了反相器脉冲输入和相应的输出，其中 t_1 和 t_2 指明了在输入和输出波形上相对应的点。

当输入为低电平时，输出就是高电平；当输入是高电平时，输出就是低电平，因此产生反相的输出脉冲。

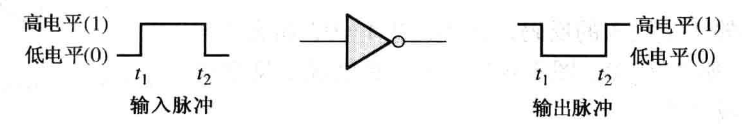


图3.2 具有脉冲输入的反相运算。打开文件F03-02检验反相操作

3.1.4 时序图

◇ 时序图给出了两个或者更多的波形在时间上的相互关系。

回顾第1章我们知道，基本上时序图是基于时间的、准确显示两个或者更多波形相互之间关系的图形。例如，图3.2中输出脉冲和输入脉冲的时间关系，可以用一个时序图把两个脉冲对准，从而使得这些脉冲边沿的发生以正确的时间关系展现出来。输入脉冲的上升沿和输出脉冲的下降沿在相同的时间出现(理想状态)。类似地，输入脉冲的下降沿和输出脉冲的上升沿在相同的时间出现(理想状态)。这种时序关系如图3.3所示。实际上，从输入的变化到输出的变化有一个非常小的延迟。时序图在说明具有多个脉冲的数字波形之间的时间关系时特别有用。

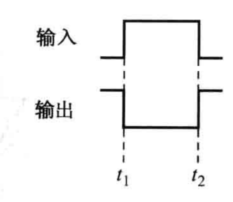


图3.3 在图3.2条件下的时序图

例3.1 图3.4中的反相器输入了一个波形。根据输入确定输出波形，并画出时序图。根据小圆圈的安放位置，有效输出状态是什么？

解： 输出波形精确地和输入波形相反(反相)，如图3.5所示，该图是基础时序图。有效或者确定的输出状态是0。

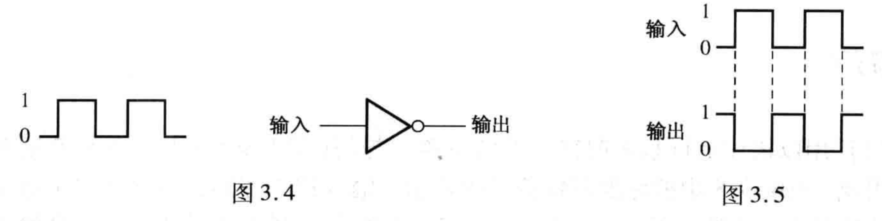


图3.4 图3.5

相关问题[①]：如果所给出的反相器否定指示(小圆圈)在输入上而不是在输出上，这会对时序图有什么影响？

3.1.5 反相器的逻辑表达式

◇ 布尔代数使用变量和运算符来描述逻辑电路。

① 答案在本章的结尾。

在布尔代数(它是逻辑电路的数学基础并且将在第4章得到全面的介绍)中，变量可以由多个字母表示，但是通常由一个或两个字母表示。接近字母表中开始位置的通常表示输入，而接近结束位置的通常表示输出。变量的反码由字母上方的横杠来表示。变量可取的值是1或者0。如果一个变量是1，它的反码就是0，反之亦然。

反相器(非门电路)的运算可以用下面的方式表示：如果输入变量为A，输出变量为X，那么

$$X = \overline{A}$$

这个表达式说明输出是输入的反码，所以如果$A=0$，那么$X=1$；而如果$A=1$，那么$X=0$。图3.6给出了这种情况。反变量$\overline{A}$可以读做A杠或者A反。

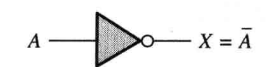


图3.6　反相器对输入变量求反

3.1.6　应用举例

图3.7给出了一个产生8位二进制数反码的电路。二进制数的位被加到反相器的输入上，二进制数的反码出现在输出中。

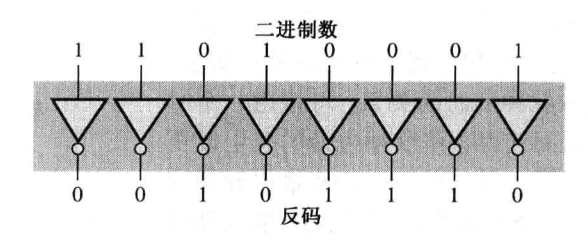


图3.7　使用反相器的反码电路举例

3.1　温故而知新　(答案在本章的结尾。)

1. 当一个反相器的输入为1时，它的输出是什么?
2. 一个反相器要求一个高电平有效脉冲(确定的输入为高电平，而不是低电平)。
   (a)针对这样的要求为反相器画出正确的逻辑符号，使用特殊形状符号和否定指示。
   (b)当正向脉冲加在反相器的输入时，给出输出。

3.2　与门

名词“门”用以描述运行基本逻辑运算的电路。与门(AND gate)由两个或者更多的输入和一个输出组成，由图3.8中的标准逻辑符号所表示。输入位于左边，而输出位于每个符号的右边。图中给出具有两个输入的与门；但是，与门可以有大于两个的任意输入。虽然给出了特殊形状符号和矩形轮廓符号，但是本书主要使用图3.8(a)中的特殊形状符号。

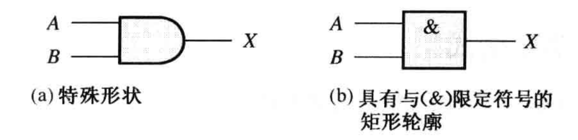


图3.8　与门的标准逻辑符号，给出了两个输入(ANSI/IEEE标准91-1984)



计算机小知识

逻辑门是计算机的基本构件。计算机中的大多数功能，除了某些特定类型的存储器，都由大规模使用的逻辑门来实现。例如，计算机的主要部件——微处理器，就是由几十万个甚至上百万个逻辑门组成。

3.2.1 与门的运算

◇ 与门可以具有多于两个的输入。

当且仅当与门所有的输入都是高电平时，才会输出高电平。当任何一个输入为低电平时，输出就是低电平。所以，与门的基本用途即判断若干条件是否同时为真，为真时所有的输入是高电平，并且在输出产生高电平以表示所有的条件都为真。图3.8中的2输入与门的输入被标记为A和B，而输出被标记为X。与门运算可以表述为

对于2输入与门，当输入A和B都是高电平时，输出X为高电平；当A或B是低电平，或者A和B都是低电平时，X就是低电平。

图3.9给出了一个2输入与门，同时列出了所有4种可能的输入组合，以及每个与门相对应的输出。

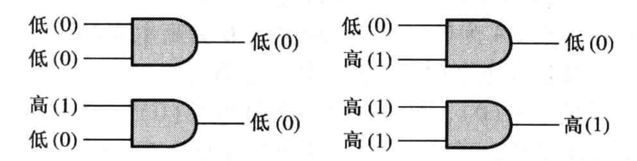


图3.9 2输入与门的所有可能逻辑电平。打开文件F03-09检验与门操作

3.2.2 与门真值表

◇ 对于一个与门来说，输入都是高电平时才会产生高电平输出。

门的逻辑运算可以用真值表来表示，真值表列出了所有的输入组合及相应的输出，如表3.2的2输入与门所示。真值表可以扩展到任意个数的输入。虽然高电平和低电平往往指“物理”意义的输入和输出状态，但是真值表给出的却是1和0；在正逻辑中，高电平就相当于1，而低电平就相当于0。对于任意的与门，不管有多少个输入，当且仅当所有的输入为高电平时，输出才是高电平。

表3.2 2输入与门的真值表

输入		输出
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

1＝高电平，0＝低电平

逻辑门输入的所有二进制组合的总数，由下面的公式确定：

$$N=2^n \tag{3.1}$$

这里N是可能输入组合的个数，n是输入变量的个数。说明如下：

2输入变量：$N=2^2=4$种组合
3输入变量：$N=2^3=8$种组合
4输入变量：$N=2^4=16$种组合

使用式(3.1)，可以为具有任意多个输入的门确定输入变量组合的个数。

例3.2 (a)为3输入与门建立真值表;(b)为4输入与门确定可能的输入变量组合的总数。

解:(a)对于3输入与门,有8个可能的输入变量的组合($2^3=8$)。真值表(见表3.3)的输入一侧给出了3位二进制数的所有8种组合。除了3个输入数位都是1的情况,输出一侧全是0。

(b)$N=2^4=\mathbf{16}$。对于4输入与门,有16种可能的二进制输入变量的组合。

相关问题:为4输入与门开发真值表。

表3.3

输入			输出
A	*B*	*C*	*X*
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

3.2.3 波形输入运算

在多数应用实例中,门的输入不是固定的电平,而是在逻辑高电平和逻辑低电平之间频繁变化的电压波形。现在来看具有脉冲波形输入的与门运算,要记住与门遵循真值表的运算,而不用考虑输入电平是不变的还是高低来回变化的电平。

检查一下与门的波形运算,通过观察相互关联的输入以确定任意给定时刻下的输出电平。在图3.10中,在时间间隔t_1期间,输入*A*和*B*都是高电平(1),因而在该期间的输出为高电平(1)。在时间间隔t_2期间,输入*A*为低电平(0)而输入*B*为高电平(1),所以输出是低电平。在时间间隔t_3期间,输入又都是高电平(1),所以输出是高电平(1)。在时间间隔t_4期间,输入*A*是高电平(1)而输入*B*是低电平(0),因此产生一个低电平(0)。最后,在时间间隔t_5期间,输入***A***是低电平(0),输入*B*是低电平(0),所以输出就是低电平(0)。正如所知,输入和输出的波形图给出了一种时间关系,称为时序图。

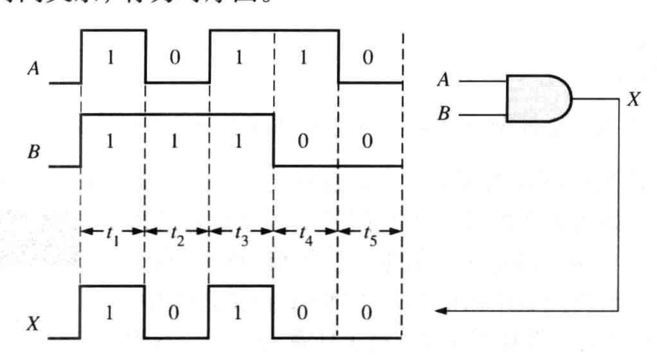


图3.10 具有时序图的与门举例,图中给出了输入和输出的关系

例3.3 如图3.11所示,如果两个输入波形*A*和*B*加到与门的输入,那么输出波形的结果是什么?

解:如图3.11中的时序图所给出的那样,当且仅当*A*和*B*的波形都是高电平时,输出波形*X*才是高电平。

相关问题:如果图3.11中波形*A*的第2个和第4个脉冲改为低电平,确定输出波形并给出时序图。

记住,在分析逻辑门的波形运算时,注意输入之间的时间关系及输入和输出之间的时间关系是非常重要的。

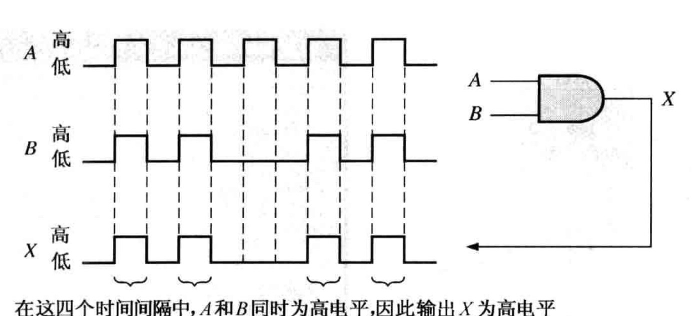


图 3.11

例 3.4 在图 3.12 中，根据两个输入波形 A 和 B，利用输出和输入之间的相应关系，给出输出波形。

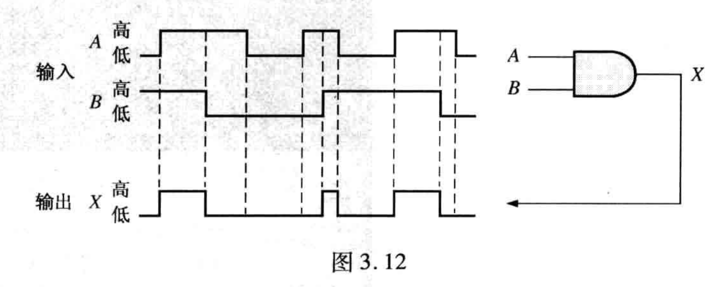


图 3.12

解： 如时序图所给出的，当两个输入波形都是高电平时，输出波形为高电平。

相关问题： 如果图 3.12 中与门的 B 输入总是高电平，确定输出波形。

例 3.5 对于图 3.13 中的 3 输入与门，确定输出波形和输入波形之间的关系。

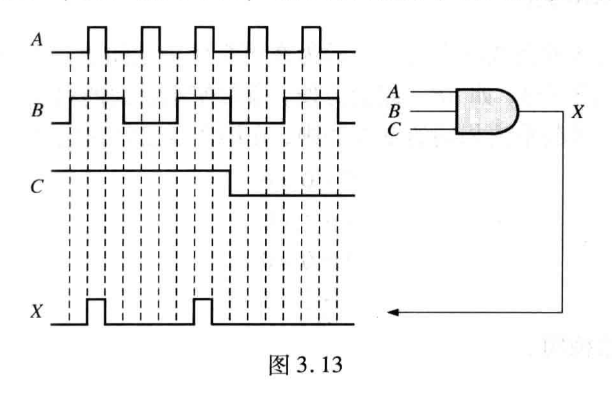


图 3.13

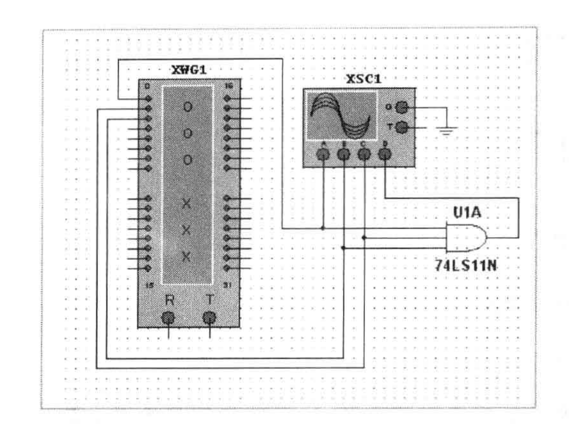
解： 当所有的 3 个输入波形 A、B 和 C 都是高电平时，3 输入与门的输出波形 X 才是高电平。

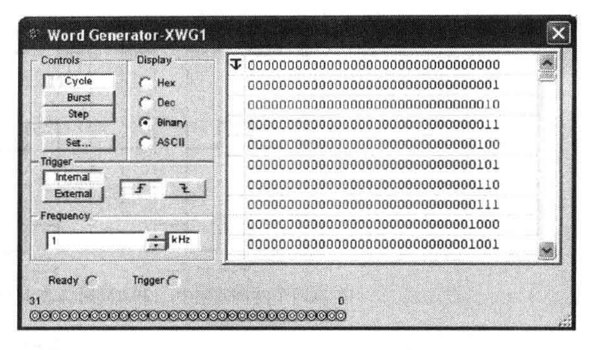
相关问题： 如果 C 输入总是高电平，那么图 3.13 中与门的输出波形是什么？

例 3.6 使用 Multisim 模拟 3 输入与门的输入和输出波形，输入波形为循环变化的二进制数 0 ~ 9。

解： 如图 3.14 所示，在加计数的模式下，使用 Multisim 的字生成器产生表示二进制序列的波形组合。示波器显示的前三个波形是输入波形，最下面的波形是输出波形。

相关问题： 使用 Multisim 软件，建立和模拟如例中所给出的 3 输入与门。





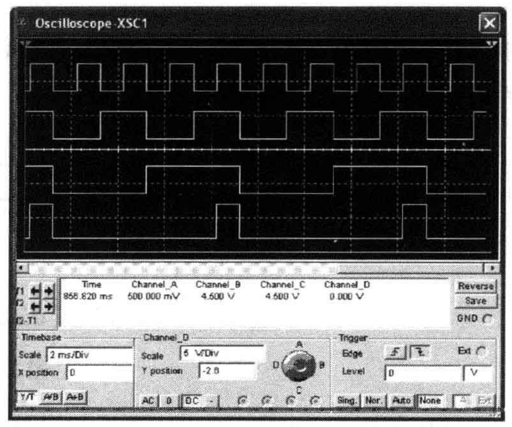


图 3.14

3.2.4 与门的逻辑表达

两个变量的逻辑与函数在数学上可以表示为在两个变量之间放一个点，如 $A \cdot B$；或不使用点，仅仅写出两个相邻的字母，如 AB。因为方便，通常使用后面的写法。

布尔乘法遵循与二进制乘法相同的基本法则，在第 2 章已经讨论过了，如下所示：

$$0 \cdot 0 = 0$$
$$0 \cdot 1 = 0$$
$$1 \cdot 0 = 0$$
$$1 \cdot 1 = 1$$

布尔乘法和与函数相同。



计算机小知识

当需要选择操作一个或者多个数据字节的某些位时，计算机可利用所有的基本逻辑运算。利用掩码可以执行选择性位操作。例如，为了清除(使之变为 0)某个数据字节的右 4 位，而保持左 4 位不变，可以把该数据字节和 11110000 进行与运算以完成这个任务。注意每一个被 0 与的位都会变成 0，而被 1 与的位保持不变。如果 10101010 和 11110000 进行与运算，结果就是 10100000。

◇ 当变量 ABC 写在一起时，表示的就是与运算。

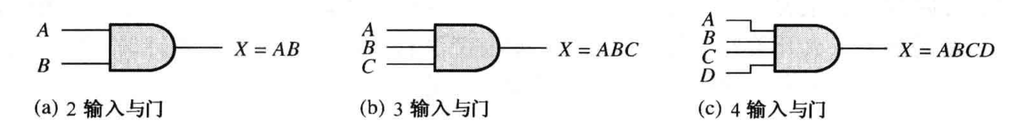
2 输入与门的运算可以表示为如下的等式形式；如果一个输入变量是 A，另一个是 B，输出

变量是 X，那么布尔表达式就是

$$X = AB$$

图3.15(a)给出了2输入变量的与门逻辑符号并标出了输出变量。

要把与表达式扩展到多于两个输入变量时，只要为每个输入变量使用一个新字母就可以了。例如，3输入与门的函数可以表示为 $X = ABC$，其中 A、B、C 是输入变量。4输入与门的表达式为 $X = ABCD$，以此类推。图3.15中的(b)和(c)分别给出了3个和4个输入变量的与门。



(a) 2输入与门 (b) 3输入与门 (c) 4输入与门

图3.15 具有2个、3个和4个输入的与门布尔表达

可以使用输出的布尔表达式来求得与门运算。例如，输入的每个变量都可以是1或0；对于2输入与门，把公式 $X = AB$ 代入就可以得到输出，如表3.4所示。这个计算表明仅当两个输入都是1(高电平)时，与门的输出 X 才是1(高电平)。可以对任意数目的输入变量做相似的分析。

表3.4

A	B	$AB = X$
0	0	$0 \cdot 0 = 0$
0	1	$0 \cdot 1 = 0$
1	0	$1 \cdot 0 = 0$
1	1	$1 \cdot 1 = 1$

3.2.5 应用举例

与门作为使能/禁止设备 与门的一种常见应用是使能(也就是允许)信号(脉冲波形)在某个时间从一点传到另一点，并禁止(阻止)在其他时间传送。

图3.16给出了与门这种特殊用途的一个简单例子，其中与门用来控制信号(波形 A)向数字计数器的传送。这个电路的目的是测量波形 A 的频率。使能脉冲有1毫秒(ms)的精确宽度。当使能脉冲是高电平时，波形 A 就会通过与门到达计数器，而当使能脉冲是低电平时，不允许(禁止)信号通过。

在使能脉冲的1 ms时间间隔内，波形 A 中的脉冲通过与门到达计数器，脉冲在1 ms的时间间隔内通过的次数等于波形 A 的频率(单位为kHz)。例如，图3.16给出了1 ms中有6个脉冲，也就是频率为6 kHz。如果在使能脉冲的1 ms时间间隔内，有1000个脉冲通过与门，这样就是1000脉冲/ms，或者为1 MHz的频率。

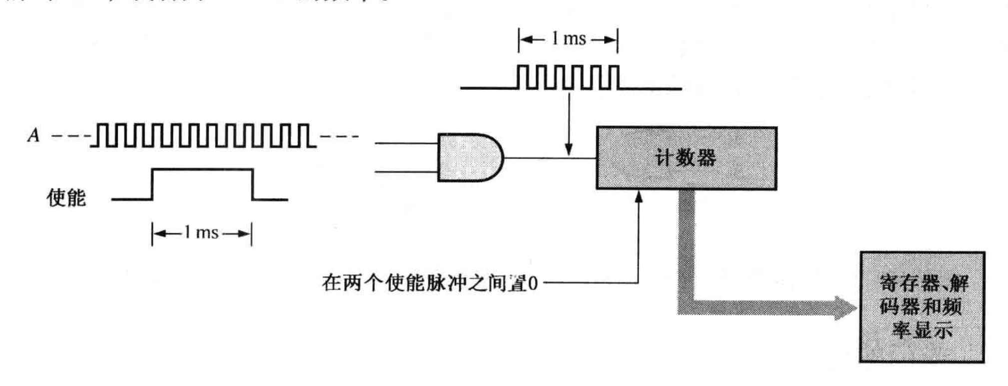


图3.16 频率计数器运行使能/禁止功能的一个与门

计数器记录每秒钟脉冲的个数，并产生一个二进制输出，进入解码和显示电路，显示可以读取的频率。使能脉冲在一定的间隔期间内重复，以产生一个新的计数值，所以如果频率变化，

就会显示出新的数值。在两个使能脉冲之间，计数器被置0，因而在每个使能脉冲到来时，计数器从0开始计数。当前的频率计数值保存在一个寄存器中，这样显示值不会受到计数器的影响。

安全带警报系统 在图3.17中，与门用在了一个简单的汽车安全带警报系统中，用来检测点火开关是否已开及安全带是否系上。如果点火开关处于打开的状态，与门的输入A上就会产生一个高电平。如果安全带没有系好，与门的输入B上就产生一个高电平。同样，当点火开关打开时，计时器就会启动并且在输入C上产生一个30秒的高电平。所有的这三个条件都存在——也就是如果点火开关处于打开的状态、安全带没有系好和计时器正在计时，这时与门的输出就是高电平，音响警报系统就会被激活以提醒司机。

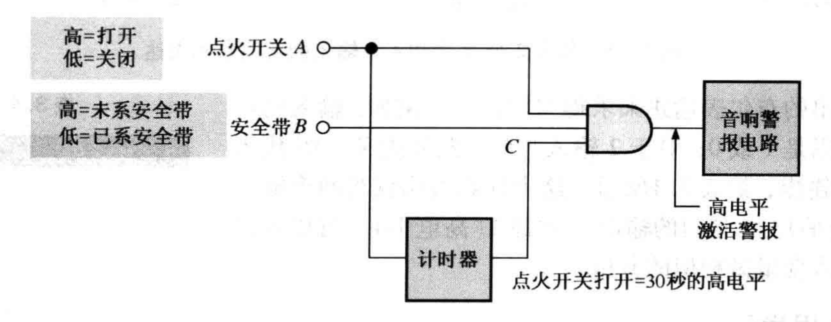


图3.17 使用与门的简单安全带警报电路

3.2节 温故而知新

1. 什么情况下与门的输出为高电平?
2. 什么情况下与门的输出为低电平?
3. 给出5输入与门的真值表。

3.3 或门

◇ 或门可以具有多于两个的输入。

或门(OR gate)具有两个或者更多的输入及一个输出，如图3.18所示的标准逻辑符号给出了具有两个输入的或门。或门可以具有多于一个的任意输入。虽然同时给出了特殊形状和矩形轮廓符号，但是本书使用的是特殊形状符号。

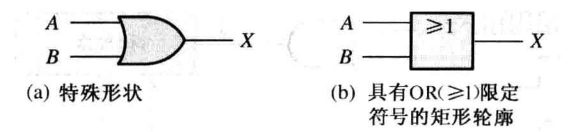


图3.18 或门的标准逻辑符号，图中给出了两个输入的或门(ANSI/IEEE标准91-1984)

3.3.1 或门的运算

◇ 有一个以上的输入是高电平时，或门的输出就是高电平。

当任意一个输入为高电平时，或门的输出就为高电平。当且仅当所有的输入是低电平时，

输出才会是低电平。所以，或门用来判断它的输入是否有一个或者多个高电平，有高电平输入时，输出一个高电平以表明条件满足。图3.18中2输入或门的输入被标以A和B，输出则被标以X，或门的运算可以做如下表述：

对于一个2输入或门来说，如果输入A和输入B中有一个是高电平，或者两者都为高电平，输出X就为高电平；仅当A和B都是低电平时，X为低电平。

高电平是或门的有效或者肯定输出电位。图3.19给出了2输入或门的所有4种可能输入组合的输入和输出结果。

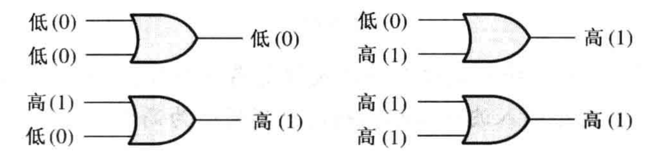


图3.19　2输入或门的所有可能逻辑电位。打开文件F03-19检验或门操作

3.3.2　或门真值表

表3.5描述了2输入或门的运算。这个真值表可以扩展到任意个数的输入；但是不管有多少输入，当有一个或者多个输入是高电平时，输出就是高电平。

表3.5　2输入或门的真值表

输入		输出
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

1 = 高电平，0 = 低电平

3.3.3　波形输入的运算

现在观察具有波形输入的或门运算，并请记住它的逻辑运算。再有，分析具有脉冲波形的或门运算的关键是涉及的所有波形的时间关系。例如，在图3.20中，在时间间隔t_1期间，A和B都是高电平1，所以输出X为高电平1。在时间间隔t_2期间，输入A是低电平0，但是输入B是高电平1，所以输出X为高电平1。在时间间隔t_3期间，两个输入都是低电平0，所以输出X为低电平0。在时间间隔t_4期间，两个输入都是高电平1，所以输出X为高电平1。

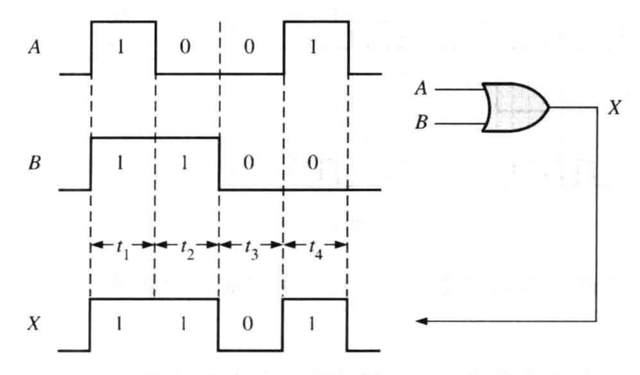


图3.20　或门运算的时序图例子，图中给出了输入和输出之间的时间关系

在这个说明中，或门的真值表运算应用到了电平没有改变的时间间隔上。例3.7到例3.9将进一步说明输入有波形的或门运算。

例3.7　在图3.21中，如果两个输入波形A和B加到或门的输入，那么输出波形是怎样的呢？

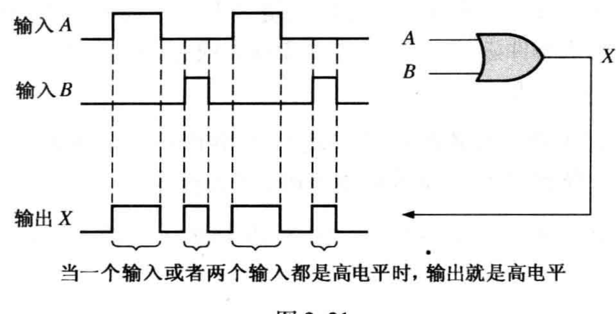


图 3.21

解：如时序图所示，当一个输入或者两个输入都是高电平时，2 输入或门的输出波形 X 就是高电平。本例的情况是，两个输入波形在相同的时刻不同时为高电平。

相关问题：如果输入 A 做如下变化：从已有的第一个脉冲的开始到第二个脉冲的结束处都为高电平，确定输出波形并给出时序图。

例 3.8 在图 3.22 中，对于两个输入波形 A 和 B，给出输出波形和输出与输入的对应关系。

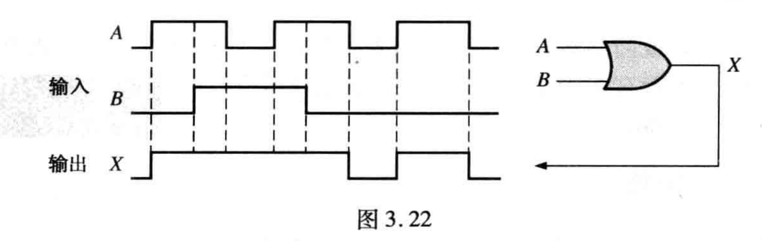


图 3.22

解：如给出的时序图中的输出波形所示，当一个或两个输入为高电平时，输出为高电平。

相关问题：如果以上输入 A 的中间脉冲由低电平取代，确定波形并画出时序图。

例 3.9 对于图 3.23 中的 3 输入或门，按照输入波形的时序关系画出其输出波形。

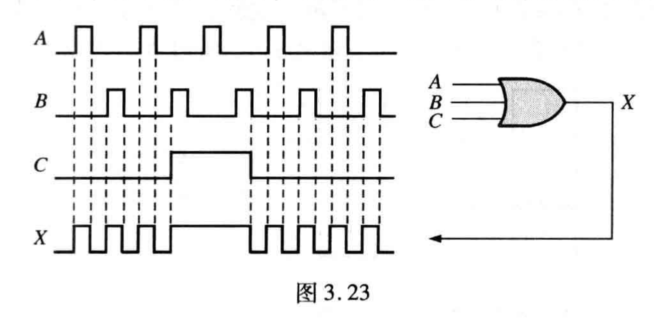


图 3.23

解：当一个输入或者两个输入都是高电平时，或门的输出就是高电平，如时序图中的输出波形 X 所示。

相关问题：如果输入 C 总是低电平，确定输出波形和时序图。

3.3.4 或门的逻辑表达式

◇ 当两个变量之间由 + 分开，那么它们是或的关系。

2 变量的逻辑或函数表示数学上两个变量的相加，例如 $A+B$。符号 + 读做“或”。

布尔代数中的加法涉及的变量是它的值仅为二进制 1 或 0。布尔代数的基本法则如下：

$$0+0=0$$
$$0+1=1$$
$$1+0=1$$
$$1+1=1$$

布尔加法和或函数相同。

注意，布尔加法和二进制加法的不同点在于两个 1 相加的情况，在布尔代数中没有进位。

2 输入或门的运算可以做如下表述：如果一个输入变量是 A，另一个输入变量是 B，输出变量是 X，那么布尔代数的表达式就是

$$X = A + B$$

图 3.24(a)给出了 2 变量的或门逻辑符号，同时标出了输入和输出变量。

(a) A, B → $X = A + B$　　(b) A, B, C → $X = A + B + C$　　(c) A, B, C, D → $X = A + B + C + D$

图 3.24　具有 2 个、3 个、4 个输入的或门布尔表达式

把或表达式扩展到多于两个的输入变量，就需要为每一个附加变量使用一个新字母。例如，3 输入或门的函数可以表示为 $X=A+B+C$，4 输入或门的表达式可以写成 $X=A+B+C+D$，以此类推。图 3.24 中的(b)和(c)分别给出了具有 3 个和 4 个变量的或门。

或门的运算可以由布尔表达式确定，把输入变量的所有 1 和 0 的可能组合代入输入变量，就可以得到输出 X，如表 3.6 的 2 输入或门所示。这个运算表明，当一个或两个输入是高电平时，输出 X 为高电平。相同的分析可以扩展到任意多个输入变量数的或门。

表 3.6

A	B	$A+B=X$
0	0	$0+0=0$
0	1	$0+1=1$
1	0	$1+0=1$
1	1	$1+1=1$

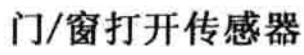


计算机小知识

另一种应用于计算机编程中的掩码运算是：有选择性地使得一个数据字节中的某些位为 1(称为置位)，而不影响任何其他的位，这是由或运算来完成的。使用掩码使得被置位的数据位的任何位置上都是 1。例如，如果想使某个字节的最高有效位等于 1，而其他位保持不变，就可以把数据字节和掩码 10000000 进行或运算。

3.3.5　应用举例

一个入室盗窃检测和警报系统的部分简化图如图 3.25 所示。这个系统可以用于一间房屋——具有两扇窗户和一扇门的房间。传感器是磁性开关，它被打开时产生一个高电平输出，关闭时产生一个低电平输出。只要窗户和门是安全的，开关就是关闭的并且或门的三个输入都是低电平。当一扇窗户或者门被打开时，在或门的输入就会产生一个高电平，这样输出就是高电平。然后激活和开启警报电路，以发出入侵警报。



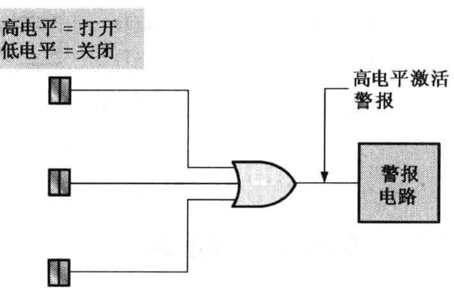


图 3.25　使用或门的一个简化入室盗窃检测和警报系统

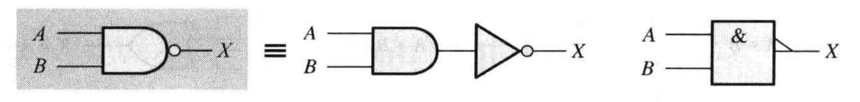
3.3 节 温故而知新

1. 什么情况下一个或门的输出为高电平?
2. 什么情况下一个或门的输出为低电平?
3. 给出一个3输入或门的真值表。

3.4 与非门

◇ 除了输出被反相之外，与非门和与门是一样的。

词汇与非(NAND)是非-与(NOT-AND)的缩写，意思是具有反码(反相)输出的与函数。2输入的与非门的标准逻辑符号和一个与门其后再加一个反相器的电路图等价，如图3.26(a)所示，其中符号≡的意思是恒等于。图3.26(b)给出了矩形轮廓符号。



(a) 特殊形状，2输入与非门及等价的与门+非门 (b) 矩形轮廓，具有极性指示

图3.26 标准与非门逻辑符号(ANSI/IEEE标准91-1984)

3.4.1 与非门的运算

只有所有的输入都是高电平时，与非门才会输出其低电平。当任何一个输入为低电平时，输出就是高电平。2输入与非门的一个具体例子如图3.26所示，其中输入被标为A和B，输出被标为X，该运算可以进行如下表述：

对于一个2输入与非门，当输入A和B都是高电平时，输出X就是低电平；当输入A或B为低电平，或者A和B都是低电平时，输出X就是高电平。

注意，这个运算和与门输出电平的运算是相反的。在一个与非门中，低电平(0)是有效或确定的输出电平，由输出的小圆圈所表示。图3.27说明了2输入与非门的所有4种可能组合，而表3.7是把2输入与非门的所有逻辑运算汇总后的真值表。

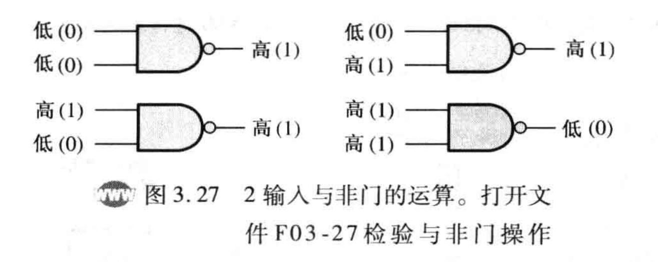


图3.27 2输入与非门的运算。打开文件F03-27检验与非门操作

表3.7 2输入与非门的真值表

输入		输出
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

1 = 高电平, 0 = 低电平

3.4.2 波形输入的运算

现在来看与非门的波形运算。从真值表可以知道，仅当所有的输入为高电平时，输出才是低电平。

例3.10 如果图3.28中的两个输入波形 A 和 B 加到与非门的输入，请确定输出波形。

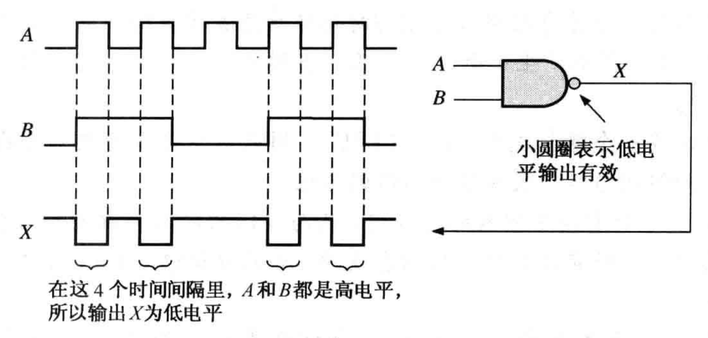


图3.28

解： 如时序图所示，只有在这个4个时间间隔里，输入波形 A 和 B 都是高电平，与非门的输出波形 X 是低电平。

相关问题： 如果输入波形 B 反相，确定输出波形并画出时序图。

例3.11 在图3.29中，按照输入波形的时序关系画出3输入与非门的输出波形。

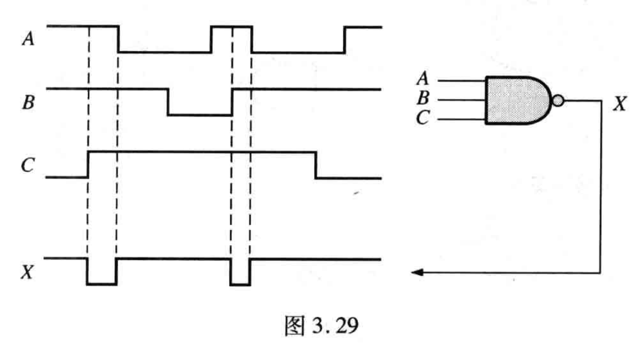


图3.29

解： 只有当所有的3个输入波形都是高电平时，与非门的输出波形 X 才是低电平，如时序图所示。

相关问题： 如果输入波形 A 反相，确定输出波形并画出时序图。

与非门的非-或等价运算 与非门运算的内在特性是这样的：一个或者多个低电平输入产生一个高电平输出。表3.7给出了当任何一个输入 A 和 B 是低电平(0)时，输出就是高电平(1)。从这个观点来看，与非门可用于需要一个或者多个低电平输入并且产生高电平输出的或运算。与非门运算的这个功能称为非-或运算。术语“非”在上下文中意思是当输入为低电平时，输入就被定义为处于有效或者确定状态。

对于2输入与非门执行非-或运算，如果输入 A 或 B 是低电平，或者 A 和 B 都是低电平时，那么输出 X 就是高电平。

当与非门用以检测一个或者多个低电平输入而不全是高电平输入时，它就是在执行非-或运算，并且由图3.30中的标准逻辑符号表示出来。虽然图3.30中的两个符号表示相同的物理门，但这是在特定的应用中用来定义的逻辑门或者运算模式，如例3.12和例3.13所示。

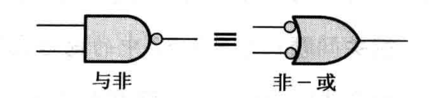


图3.30 表示与非门的两个等效运算的标准符号

例3.12 一个制造工厂使用两个存储罐保存在制造加工过程中所需要的某种液体化学物质。每一个存储罐都有一个传感器用来检测何时化学物质液位会降到满罐的25%。当存储罐储量大于1/4满罐时，传感器就产生一个5 V电压。当罐中的化学物质储量降至1/4满罐时，传感器就会输出0 V电压。

指示器面板上需要一个绿色发光二极管(LED)，用来显示什么时候两个存储罐的储量都大于1/4满罐。给出如何用与非门实现这个功能的方法。

解：图3.31给出了具有两个输入的与非门，此与非门的输入和存储罐液位传感器连接，输出连接到指示器面板上。所完成的功能可以表述为：如果存储罐A和B的储量都达到1/4满罐以上，LED就会点亮。

只要两个传感器输出都是高电平(5 V)，也就是表示两个存储罐的储量都大于1/4满罐，那么与非门的输出就是低电平(0 V)。设计绿色LED电路使得低电平点亮。

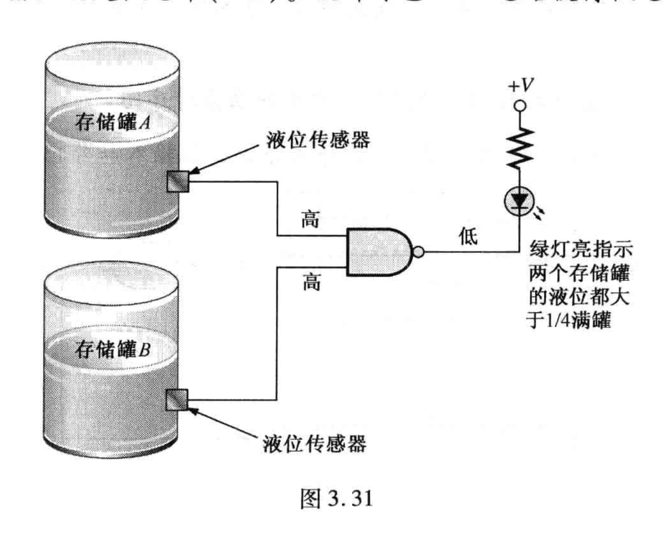


图3.31

相关问题：若要监测3个存储罐而不是2个存储罐，应当对图3.31中的电路做怎样的修改?

例3.13 例3.12中所描述的生产过程的管理员，决定使用一个红色LED显示下面情况：只要有一个存储罐的液位降至1/4满罐时就点亮，替代绿色LED点亮指示两个存储罐液位都高于1/4满罐的情况。给出如何实现这个功能的方法。

解：图3.32给出了一个非-或门符号的与非门，用以检测至少有一个低电平输入的情况。如果存储罐内体积达到1/4满罐或更低时，传感器就输出一个低电平电压。发生这种情况时，非-或门就会输出高电平，这个高电平使得面板上的电路把红色LED点亮。所完成的功能可以表述为：如果存储罐A或B或者A和B同时低于1/4满罐，LED就会点亮。

注意：这个例子和例3.12使用了相同的2输入与非门，但是在图表中使用了不同的符号，说明了与非门和等效的非-或门运算在不同方式下的使用。

相关问题：对图3.32中的电路怎样进行修改，才能监测4个存储罐而不是2个存储罐?

例3.14 对于图3.33中的4输入与非门，这里看做非-或门的运算，根据输入确定输出。

解：只要有一个输入是低电平，输出波形X就是高电平，如时序图所示。

相关问题：如果输入波形A在应用于与非门之前被反相，确定输出波形。

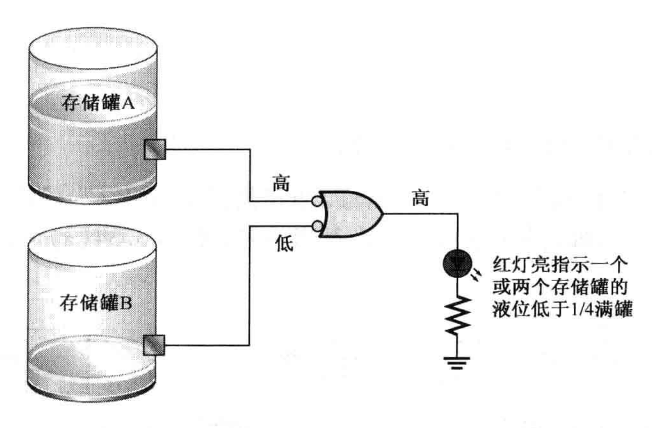


图 3.32

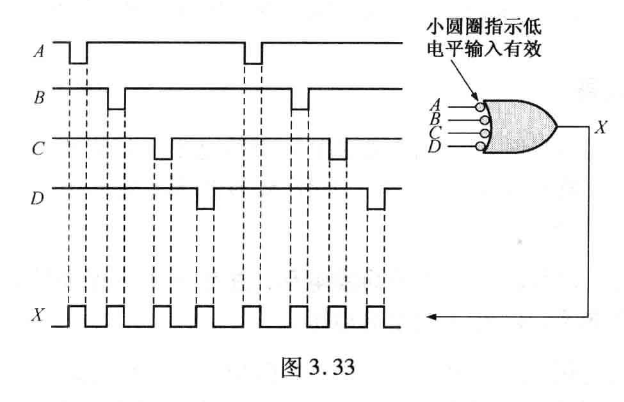


图 3.33

3.4.3 与非门的逻辑表达式

◇ 变量上方的横杠就表示反相。

2 输入与非门输出的布尔表达式为

$$X = \overline{AB}$$

这个表达式指出，对于两个变量 A 和 B，首先进行与运算然后再取反，这由与运算表达式上的横杠来表示。这就是 2 输入与非门运算的方程表示形式。两个输入变量所有可能的值由此方程可以得到输出，结果如表 3.8 所给出。

一旦给定的逻辑函数的表达式被确定下来，对应于变量所有可能的值，通过函数都可以得出运算结果。对于每一种输入组合，运算结果会准确地给出逻辑电路的输出，因此就对这个电路的逻辑运算做出了完整的描述。与非门表达式可以扩展到多于两个的输入变量，可以添加其他的字母来表示扩展的变量。

表 3.8

A	B	$\overline{AB} = X$
0	0	$\overline{0 \cdot 0} = \overline{0} = 1$
0	1	$\overline{0 \cdot 1} = \overline{0} = 1$
1	0	$\overline{1 \cdot 0} = \overline{0} = 1$
1	1	$\overline{1 \cdot 1} = \overline{1} = 0$

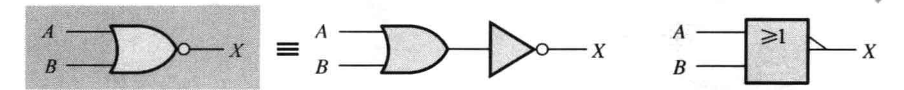
3.4 节 温故而知新

1. 在什么情况下与非门的输出为低电平？
2. 在什么情况下与非门的输出为高电平？
3. 描述与非门和非-或门的差别？它们是否有相同的真值表？
4. 输入为 A、B 和 C，写出与非门的输出表达式。

3.5 或非门

◇ 除了输出反相之外，或非门和或门相同。

术语或非(NOR)是非-或(NOT-OR)的缩写，意指具有反相(反码)输出的或运算。2 输入或非门和与之等价的或门后面再加反相器的标准逻辑符号，如图 3.34(a)所示。图 3.34(b)给出了矩形轮廓符号。



(a)特殊形状，2 输入或非门和等价的非-或门　　(b)矩形轮廓，具有极性指示的 2 输入或非门

图 3.34　标准或非门逻辑符号(ANSI/IEEE 标准 91-1984)

3.5.1 或非门的运算

当任何一个输入为高电平时，或非门就输出一个低电平。只有当所有的输入是低电平时，输出才是高电平。对于 2 输入或非门的情况，如图 3.34 所示，输入被标为 A 和 B，输出被标记为 X，该运算可以做如下表述：

对于 2 输入或非门，如果输入 A 或 B 为高电平，或者 A 和 B 都是高电平，输出 X 就是低电平；如果 A 和 B 都是低电平，输出 X 就是高电平。

这个运算产生和或门相反的输出电平。在或非门中，低电平输出是有效的或者是确定的输出电平，由输出的小圆圈所表示。图 3.35 说明了 2 输入或非门的运算，包括所有可能的 4 种输入组合，而表 3.9 是 2 输入或非门的真值表。

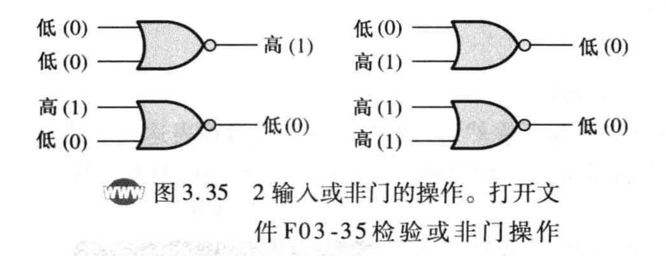


图 3.35　2 输入或非门的操作。打开文件 F03-35 检验或非门操作

表 3.9　2 输入或非门的真值表

输入		输出
A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

1 = 高电平，0 = 低电平

3.5.2 波形输入的运算

下面的两个例子给出了具有脉冲波形输入的或非门运算。和其他类型的门一样，只要遵循真值表就可以确定输出波形和输入的正确时间关系。

例 3.15　如果图 3.36 中的两个波形加到或非门的输入，那么输出波形是什么？

解： 无论什么时候只要或非门的输入为高电平，输出就是低电平，如时序图中的输出波形 X 所示。

相关问题： 把输入 B 反相，确定输出波形和输入波形之间的关系。

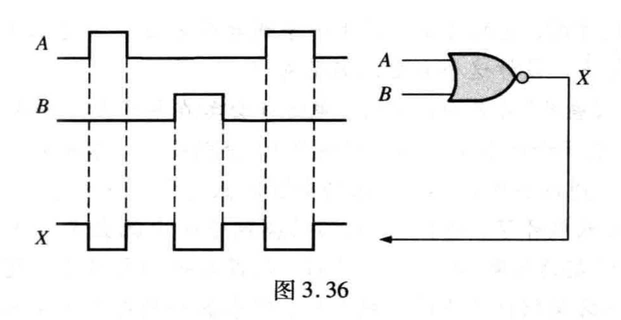


图 3.36

例 3.16 给出图 3.37 中 3 输入或非门的输出波形和输入波形之间的正确时间关系。

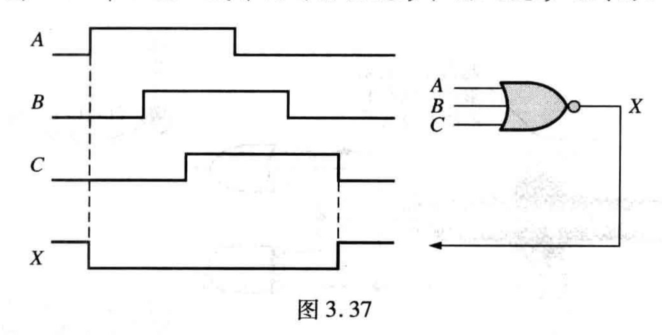


图 3.37

解： 当任何一个输入为高电平时，输出 X 就是低电平，如时序图中的输出波形 X 所示。

相关问题： 在输入 B 和 C 反相的情况下，确定输出并画出时序图。

或非门的非-与等价运算 或非门和与非门相似，但具有其运算的另一个方面，也就是固有的逻辑函数功能。表 3.9 给出了只有当所有的输入都是低电平时，才会产生高电平输出。从这一点来说，或非门可以用做需要所有的低电平输入来产生高电平输出的与运算。或非门的这个功能称为非-与。这里术语“非”意指低电平输入有效或低电平为确定状态。

对非-与运算的 2 输入或非门来说，如果输入 *A* 和 *B* 都是低电平时，输出 *X* 就是高电平。

当一个或非门用来检测所有的低电平输入而不是一个或者多个高电平输入时，它就是在执行非-与运算，这由图 3.38 中的标准符号表示。图 3.38中的两个符号表示相同的物理门并且仅用于区分两种运算模式，记住这一点很重要。下面的三个例子给出了说明。

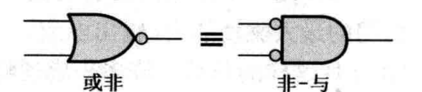


图 3.38 表示或非门两种等价运算的标准符号

例 3.17 当两个低电平出现在输入并且输出高电平时，就需要一个设备进行指示，给出此设备。

解： 当两个输入都是低电平并且输出为高电平时，就需要使用 2 输入或非门，并把它看成非-与门，如图 3.39 所示。

相关问题： 当有一个或者两个高电平输入并且产生低电平输出结果时，就需要一个设备进行指示，给出这个设备。

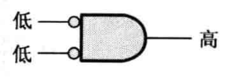


图 3.39

例 3.18 作为飞机功能监测系统的一部分，需要一个电路来指示着陆之前起落架的状态。当准备着陆时，“起落架展开”开关打开，这时如果所有的三个起落架都正确展开，绿色 LED 就会点亮。如果着陆前有任何一个起落架没有正确展开，那么红色

LED 被点亮。起落架展开时，它的传感器产生一个低电平电压。当起落架处在收回状态时，传感器就会产生一个高电平。设计这个电路以满足需求。

解： 只有在“起落架展开”开关被打开时，电源才会加在电路上。如图 3.40 所示，对应于以上的两种情况(起落架展开和起落架没有正确展开)，分别使用一个或非门以满足要求。其中一个或非门用做非-与门，用以检测三个起落架传感器的低电平。当所有三个门都是低电平，也就是这三个起落架都被正确展开了，那么非-与门的高电平输出就会点亮绿色 LED。另一个或非门运行或运算，检测当“起落架展开”开关打开时，是否有一个或者多个起落架仍然处于收回状态。当一个或者多个起落架仍然处于收回状态时，传感器的高电平就会传到或非门，从而产生一个低电平输出来点亮红色 LED。

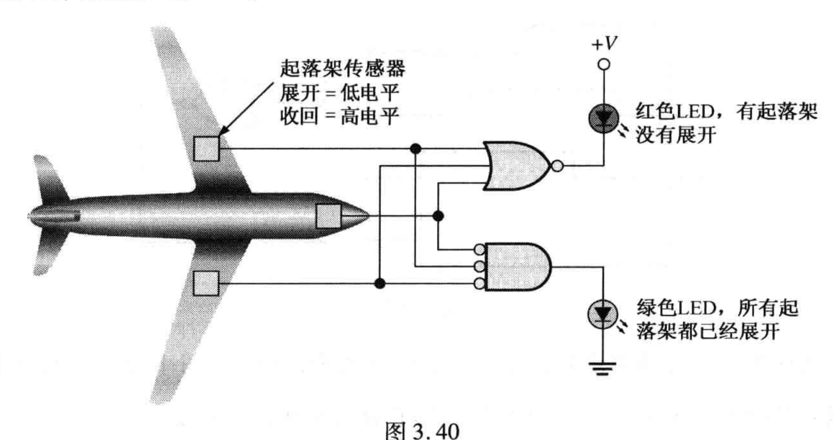


图 3.40

相关问题： 在飞机起飞之后，应当使用什么类型的门来检测是否所有的三个起落架都已收回，假设需要一个低电平输出去点亮 LED。



应用提示

当使用一个逻辑门去驱动如 LED 这样的负载时，查询一下生产商的数据资料，以得到逻辑门的最大驱动能力(输出电流)。通常一个集成逻辑门可能不能输出足够的电流，从而驱动如 LED 这样的负载。许多集成逻辑门带有缓冲输出，如集电极开路或漏极开路输出。典型的集成逻辑门的输出电流大小在微安(μA)或低于毫安(mA)数量级，例如，标准的 TTL(晶体管-晶体管逻辑电路)的输出电流可以达到 16 mA。大多数 LED 所需要的电流在 10 ~ 50 mA。

例 3.19 对于图 3.41 中执行非-与运算的 4 输入或非门，确定与输入相对应的输出。

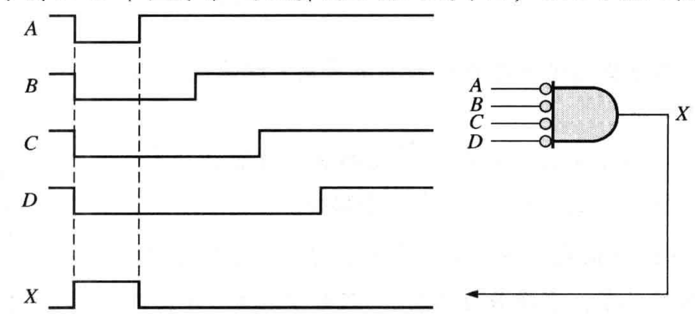


图 3.41

解： 当所有的输入都是低电平时，输出才会是高电平，如时序图中的输出波形 X 所示。

相关问题： 确定输入 D 反相后的输出，并画出时序图。

3.5.3 或非门的逻辑表达式

2 输入或非门输出的布尔表达式可以写为

$$X = \overline{A + B}$$

这个等式说明，两个输入变量先进行或运算，然后再求反，求反由或运算表达式上方的横杠指示。运算这个表达式，就可以得到如表 3.10 所示的结果。该或非运算表达式可以扩展到多于两个的输入变量，使用另外的字母表示其他变量即可。

表 3.10

A	B	$\overline{A+B} = X$
0	0	$\overline{0+0} = \overline{0} = 1$
0	1	$\overline{0+1} = \overline{1} = 0$
1	0	$\overline{1+0} = \overline{1} = 0$
1	1	$\overline{1+1} = \overline{1} = 0$

3.5 节 温故而知新

1. 在什么情况下或非门的输出为高电平？
2. 在什么情况下或非门的输出为低电平？
3. 描述或非门和非-与门两者功能的不同之处。它们是否有相同的真值表？
4. 输入变量为 A、B 和 C，写出 3 输入或非门的输出表达式。

3.6 异或门和同或门

3.6.1 异或门

异或门(XOR)的标准符号如图 3.42 所示。异或门只有两个输入。异或门的运算功能为模-2 加法(第 2 章介绍过)。只有当两个输入处于相反的逻辑电平时，异或门的输出才是高电平。根据输入 A 和输入 B 及输出 X，它的运算可以做如下表述：



计算机小知识

异或门连接起来构成一个加法电路，可以在计算机的算术逻辑单元(ALU)中执行加法、减法、乘法及除法。异或门由基本逻辑门与门、或门和非门组成。

对于异或门来说，如果输入 A 是低电平而输入 B 是高电平，或者输入 A 是高电平而输入 B 是低电平，那么输出 X 就是高电平；如果 A 和 B 都是高电平或者都是低电平，那么输出 X 为低电平。

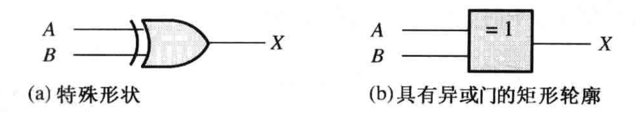


图 3.42 异或门的标准逻辑符号

◇ 对于异或门来说，相反的两个输入产生高电平输出。

异或门的所有 4 个可能的输入组合及输出结果如图 3.43 所示。高电平是有效或者是确定电平，只有当两个输入相反时才会输出高电平。异或门的运算总结在表 3.11 中。

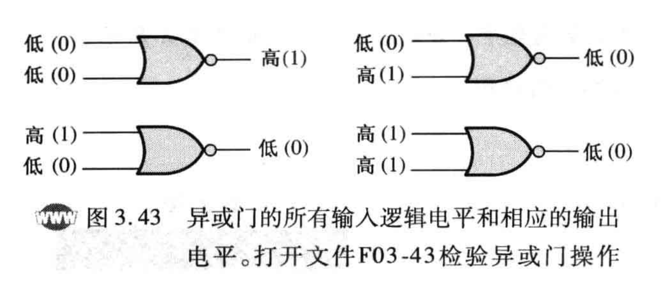


图 3.43 异或门的所有输入逻辑电平和相应的输出电平。打开文件F03-43检验异或门操作

表 3.11 异或门的真值表

输入 A	输入 B	输出 X
0	0	0
0	1	1
1	0	1
1	1	0

例 3.20 一个系统包含两个并行运行的相同电路。只要它们正确运行，两个电路的输出就总是相同的。如其中一个电路没有正确运行，那么同时就会有相反的输出电平。设计一个方法用以检测这两个电路中的一个出现了故障。

解： 如图 3.44 所示，电路的输出连接异或门的输入。任何一个电路出现故障都会产生不同的输出，使得异或门的两个输入为相反的电平。这样就会在异或门的输出产生一个高电平，以指示其中的一个电路出现了故障。

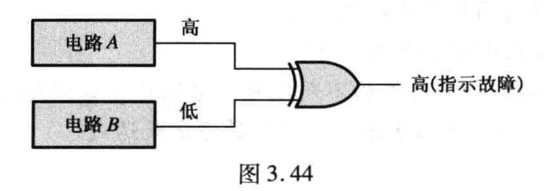


图 3.44

相关问题： 异或门总是检测图 3.44 中两个电路同时发生的故障吗？如果不是，应当处于什么条件？

3.6.2 同或门

同或门(XNOR)的标准符号如图 3.45 所示。和异或门相似，同或门也只有两个输入。同或门符号输出位置上的小圆圈指示它的输出和异或门的输出是相反的。当两个输入逻辑电平相反时，同或门的输出就是低电平。这个运算可以做如下表述(A 和 B 是输入，X 是输出)：

对于同或门来说，如果输入 A 是低电平而输入 B 是高电平，或者 A 是高电平，B 是低电平，输出 X 就是低电平；如果输入 A 和输入 B 都是高电平或者都是低电平，输出 X 就是高电平。

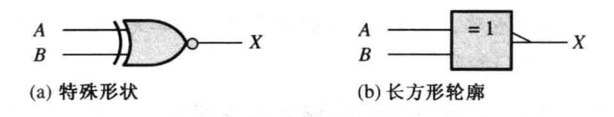


图 3.45 同或门的标准逻辑符号

同或门的 4 个可能输入组合及输出结果如图 3.46 所示。同或门的运算总结在表 3.12 中。注意当两个输入具有相同的电平时，输出就是高电平。

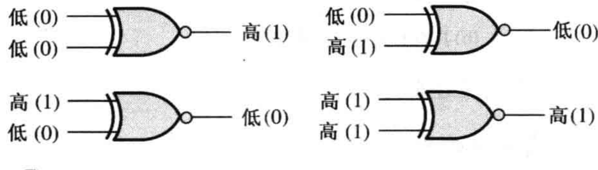


图 3.46 同或门的所有输入逻辑电平和相应的输出电平。打开文件F03-46检验同或门操作

表 3.12 同或门的真值表

输入 A	输入 B	输出 X
0	0	1
0	1	0
1	0	0
1	1	1

3.6.3 波形输入的运算

正如对其他门所做的运算一样，观察异或门和同或门在脉冲波形输入条件下的运算。和以前一样，在脉冲波形输入的每个特定时间段中，对如图3.47所示的异或门应用真值表进行运算。可以看到在时间段 t_2 和 t_4 期间，输入波形 A 和 B 为相反的电平。所以在这两个时间段，输出 X 是高电平。由于在时间段 t_1 和 t_3 期间，两个输入为相同电平，要么都是高电平要么都是低电平，所以这两个时间段的输出 X 是低电平，如时序图所示。

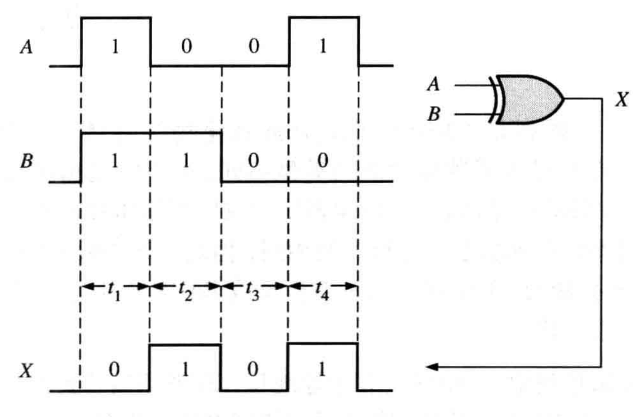


图3.47 异或门脉冲波形输入的运算

例3.21 在图3.48中，给定了输入波形 A 和 B，确定异或门和同或门的输出波形。

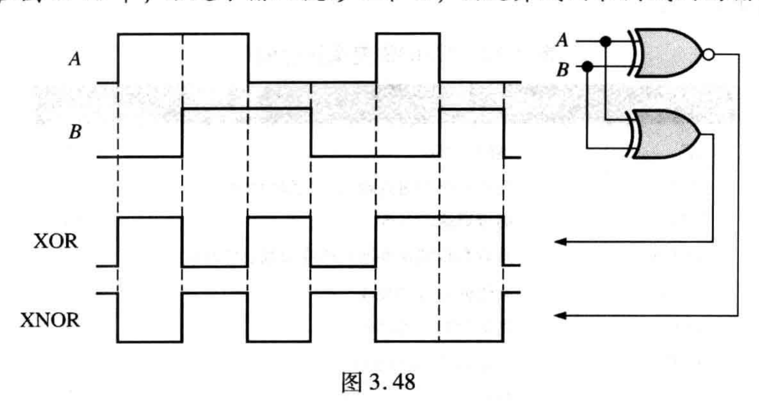


图3.48

解： 输出波形如图3.48所示。注意只有当异或门的两个输入相反时，输出才是高电平。只有当同或门的两个输入相同时，输出才是高电平。

相关问题： 如果把以上两个输入波形 A 和 B 反相，请确定输出波形。

3.6.4 应用举例

异或门可以用做2位模-2加法器。在第2章中，二进制加法的基本规则是：0+0=0，0+1=1，1+0=1，以及1+1=10。观察异或门的真值表，就会发现输出是两个输入的和运算。在这种情况下，两个输入都是1，输出的和是0，但是我们丢弃了进位1。在第6章中，将会看到异或门怎样组合在一起形成完整的加法电路。表3.13说明了异或门用做模-2加法的运算，将其用在循环冗余校验码(CRC)系统中，以完成第2章描述的除法运算。

3.6 节 温故而知新

1. 什么情况下异或门的输出为高电平?
2. 什么情况下同或门的输出为高电平?
3. 当两个位不相同时,如何用异或门来检测?

表 3.13 异或门用做两位加法

输入位		输出(和)
A	B	Σ
0	0	0
0	1	1
1	0	1
1	1	0 (没有进位1)

3.7 固定功能逻辑

3.7.1 逻辑系列

数字逻辑电路的三个系列是 CMOS(互补金属氧化物半导体)、双极型和 BiCMOS(双CMOS)。这三种电路器件实现内部逻辑功能的类型不同。CMOS 是由场效应管(FET)来实现,双极型(也称为 TTL,即晶体管-晶体管逻辑)逻辑使用双极型晶体管来实现,而 BiCMOS 是前面两种的组合。这些逻辑器件系列的基本逻辑功能相同,但是工作参数不同,例如开关速度(传输时间)、功率损耗和抗干扰能力。CMOS 与门和双极型或 BiCMOS 与门的逻辑功能相同。对于所有其他功能的逻辑器件也一样。

CMOS 在这些逻辑系列中,CMOS 占有主导地位,双极型显然要被淘汰,BiCMOS 具有的逻辑功能类型也有限。在 CMOS 系列中,按不同的使用电压、功率损耗、开关速度和一些其他参数分为许多种类。表 3.14 列出了更多的常用 CMOS 逻辑电路,按各种器件的电路运行和优化的直流工作电压进行区分。

表 3.14 CMOS 逻辑器件的种类

名称	描述	V_{CC}
AC	高级CMOS	5.0 V
ACT	具有双极型兼容输入的高级CMOS	5.0 V
AHC	高级高速CMOS	5.0 V
AHCT	具有双极型兼容输入的高级高速CMOS	5.0 V
ALVC	高级低电压CMOS	3.3 V
AUC	高级超低电压CMOS	1.8 V
AUP	高级超低功耗CMOS	3.3 V
AVC	高级甚低电压CMOS	2.5 V
CD4000	标准CMOS	5.0 V
FCT	快速CMOS技术	5.0 V
HC	高速CMOS	5.0 V
HCT	具有双极型兼容输入的高速CMOS	5.0 V
LV-A	低电压CMOS	3.3 V
LV-AT	具有双极型兼容输入的低电压CMOS	5.0 V
LVC	低电压CMOS	3.3 V

双极型 如同 CMOS 那样,双极型逻辑器件系列也有不同参数的种类。这些种类列在表 3.15 中,所有的不同种类都运行于典型的 5 V 电压 V_{CC}下。

BiCMOS 表 3.16 给出了 BiCMOS 逻辑系列中的常用种类。工作电压 V_{CC}都是典型的 5 V。

表 3.15　双极型逻辑器件的种类

名　称	描　述
ALS	高级低功率肖特基(Schottky)
AS	高级肖特基
F	高速
LS	低功耗肖特基
S	肖特基
None	标准TTL

表 3.16　BiCMOS 逻辑器件的种类

名　称	描　述
ABT	高级BiCMOS
ALB	高级低电压BiCMOS
BCT	标准BiCMOS
LVT	低电压BiCMOS

3.7.2　逻辑门

所有的逻辑功能——非、与、或、与非、或非、异或和同或，都可以在 CMOS 和双极型中找到。除此之外，需要大电流驱动负载的缓冲输出门也有相应的逻辑功能。典型的集成电路(IC)封装具有的门的配置类型在系列的定义中由最后两位或三位数字进行区别。例如，74LS04 是一块低功耗肖特基十六反相器。一些常用的逻辑门的配置和它们的标准鉴别数字如下所示：

- 四 2 输入与非门—**00**
- 四 2 输入或非门—**02**
- 十六反相器—**04**
- 四 2 输入与门—**08**
- 三 3 输入与非门—**10**
- 三 3 输入与门—**11**
- 四同或门—**266**
- 四 4 输入与非门—**20**
- 四 2 输入与门—**21**
- 三 3 输入或非门—**27**
- 单 8 输入与非门—**30**
- 四 2 输入或门—**32**
- 四异或门—**86**

IC 封装　所有 74 系列的 CMOS 和相同类型的双极型器件的引脚是兼容的。这就意味着 CMOS 数字 IC 芯片和相对应的双极型芯片的输入/输出引脚相同，例如 74HC00(四 2 输入与非门)，即在一块 IC 芯片上有 4 个 2 输入的与非门。对于典型的 IC 封装，双引线封装(DIP)用于插塞或穿通安装，小轮廓集成电路(SOIC)封装用于表面安装，如图 3.49 所示。在某些情况下还有其他类型的封装，SOIC 比 DIP 明显小得多。以上列出的大多数固定功能的逻辑芯片的引脚配置如图 3.50 所示。

单个门的逻辑　CMOS 单个门的封装种类很有限。对于只有一个门的封装，该系列形成有 5 个极小引脚的封装，往往用于一些特殊的场合，例如用于一些紧凑的、可用空间受到限制的地方，需要把这样的逻辑芯片挤进去以便达到最终的更改效果。

逻辑符号　固定功能集成电路的逻辑符号使用标准的门的符号，IC 封装给出了芯片中门的数量及与每个门相关联的引脚编号，也给出了电源 V_{CC} 和地线的引脚编号。图 3.51 给出了一个十六反相器和四 2 输入与非门的例子，其中给出特殊形状符号和矩形轮廓符号两种形式的图。不考虑逻辑系列，所有的芯片都有相同的前缀，也就是它们有相同的引脚排列。例如，7400、74S00、74LS00、74ALS00、74F00、74HC00 和 74AHC00 都是四 2 输入与非门芯片，它们的引脚都兼容。

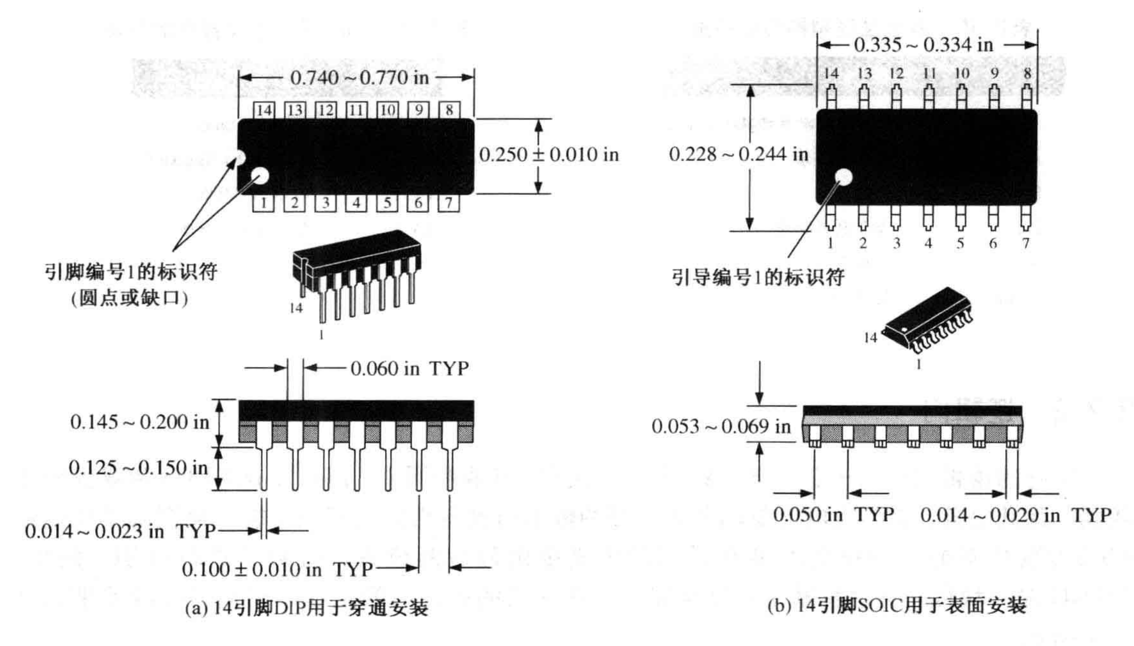


图 3.49　典型的 DIP 和 SOIC 封装的引脚编号和基本尺寸的示意图

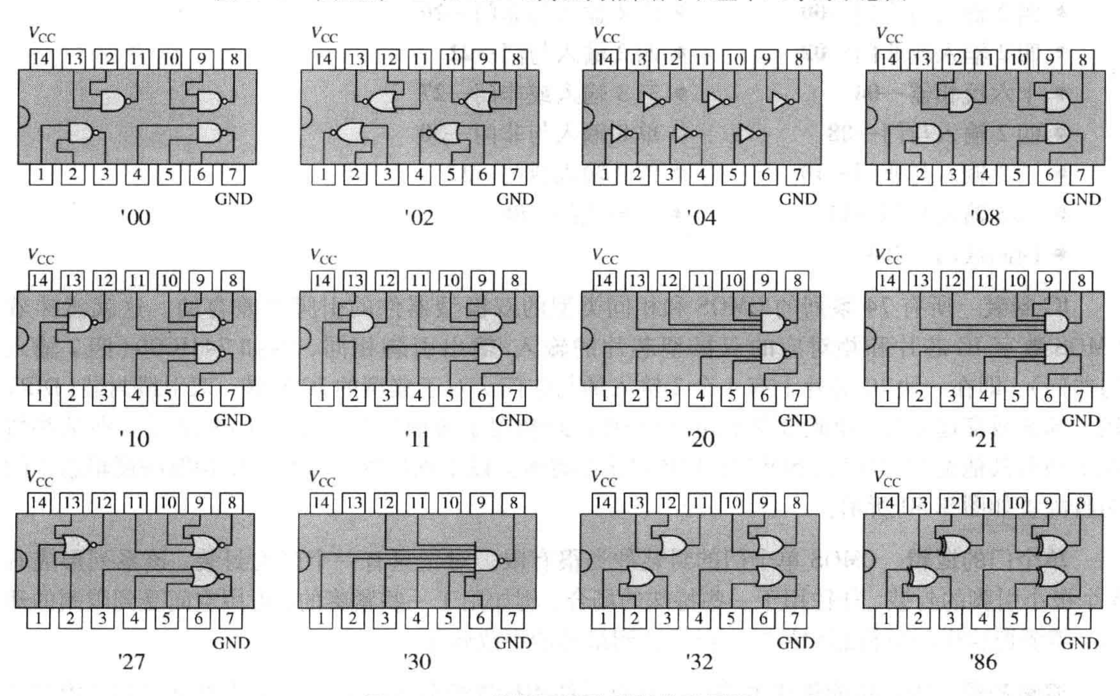


图 3.50　一些常用门及引脚的配置图

应用提示：使用 CMOS 的注意事项

CMOS 逻辑芯片对静电非常敏感，如果操作不当，CMOS 会由于静电放电(ESD)而损坏。如下为主要注意事项：

1. 保存和安放在导电泡沫上。
2. 把仪器和地连接上。
3. 手腕串联一个大电阻和地连接。
4. 在电源没有关上时，不要把芯片拔掉。
5. 电源关上时，不要加信号电压。

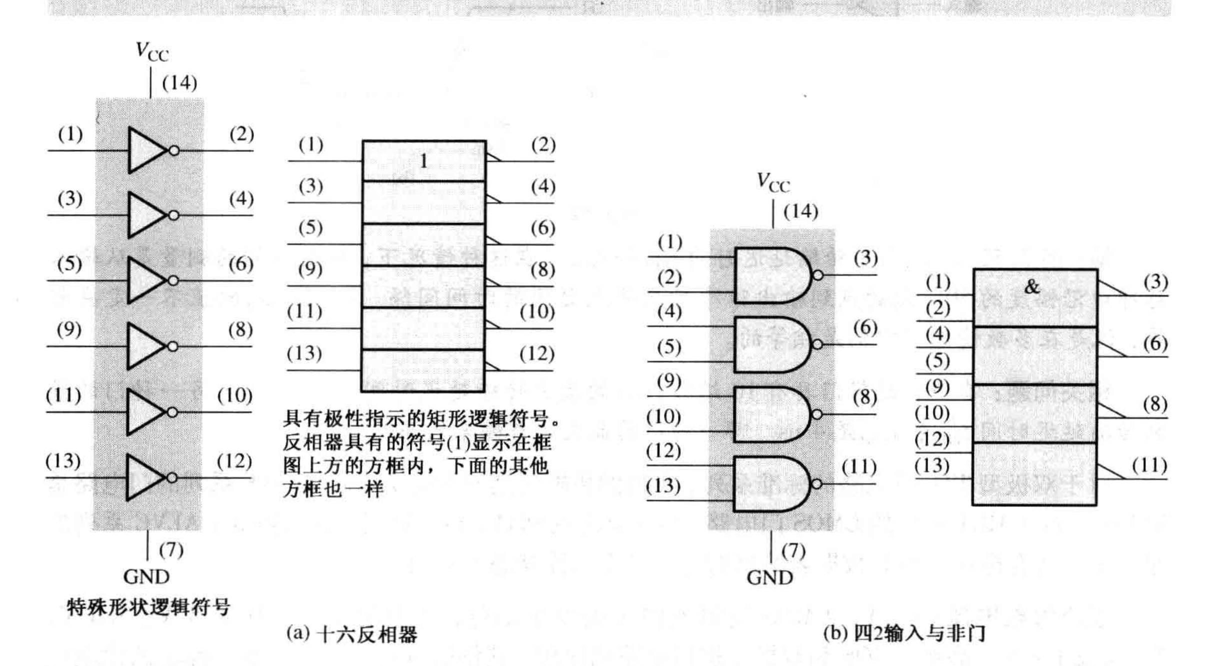


图3.51 十六反相器的逻辑符号(后缀04)和四2输入与非门(后缀00)。符号适用于任何CMOS或双极型系列的相同设备

3.7.3 工作特性和参数

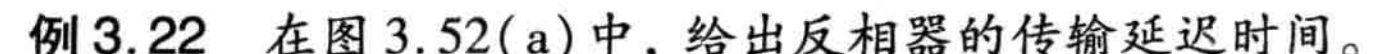
◇ 高速逻辑芯片具有很小的传输延迟时间。

定义逻辑电路的几个工作特性，它们是按照传输延迟时间测量得到的开关速度、功率损耗、扇出或驱动能力、速度-功率乘积、直流供电电压和输入/输出逻辑电平。

传输延迟时间 此参数是在逻辑电路能够工作的最大开关速度或频率下得到的。对于逻辑电路，低速和高速是相对于传输延迟时间而言的。传输延迟时间越短，电路的速度就越高，工作频率也越高。

传输延迟时间 t_p，即逻辑门的 t_p 是输入脉冲的出现到此脉冲在输出出现的时间间隔。有两种测量与一个逻辑门有关的传输延迟时间的方法，它们适用于所有其他种类的基本门电路。

- t_{PHL}：输入脉冲的特定参考点和对应传输得到的输出脉冲参考点之间的时间，如输出从高电平变化到低电平(HL)的点。
- t_{PLH}：输入脉冲的特定参考点和对应传输得到的输出脉冲参考点之间的时间，如输出从低电平变化到高电平(LH)的点。



例 3.22 在图 3.52(a)中，给出反相器的传输延迟时间。

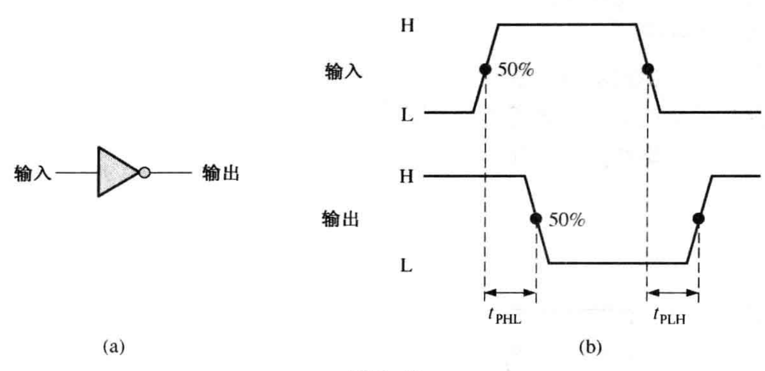


图 3.52

解： 图 3.52(b)给出了传输延迟时间 t_{PHL} 和 t_{PLH}。在这种情况下，延迟时间的测量是从输入脉冲边沿幅度的50% 处的点到输出脉冲对应的点之间的时间间隔。t_{PHL} 和 t_{PLH} 的值不一定要相等，但是在多数情况下它们是相等的。

相关问题： 有一种逻辑门具有 10 纳秒(ns)的最大传输延迟时间 t_{PHL} 和 t_{PLH}。另一种门的最大传输延迟时间 t_{PHL} 和 t_{PLH} 是 4 ns。哪一种门的最大工作频率较高？

对于双极型 TTL 门电路的标准系列，典型的传输延迟时间是 11 ns，对于 F 系列的门电路是 3.3 ns。对于 HCT 系列的 CMOS 门电路，传输延迟时间是 7 ns；AC 系列的是5 ns；ALVC 系列的是 3 ns。所有标定的值如数据表给出的是在某个工作状态下的值。

直流供电电压(V_{CC}) CMOS 逻辑电路按类型来划分，典型的供电电压是 5 V、3.3 V、2.5 V或 1.8 V。高级 CMOS 和双极型逻辑电路相比较，其供电电压的范围很宽。标定的供电电压为5 V的 CMOS 电路可以允许 2 ~6 V 的供电电压，尽管电压的大小对传输延迟时间和功率损耗的影响很大，但是在此电压范围内仍然可以正常工作。标定的供电电压为 3.3 V 的 CMOS 电路可以工作在 2 ~3.6 V 供电电压下。对于典型的双极型逻辑电路，典型的直流供电电压是 5 V，可以工作在 4.5 ~5.5 V 的供电电压下。

◇ 更低的功率损耗意味着从直流供电电压中获取更小的电流。

功率损耗 一个逻辑门的功率损耗 P_D，是直流供电电压和平均工作电流的乘积。通常情况下，门电路的输出为低电平时的工作电流比它是高电平时要低。生产商的数据表通常给出的是输出为低电平下的工作电流，即为 I_{CCL}，如为高电平，那么为 I_{CCH}。平均供电电流基于占空比 50% 来确定(输出低电平和高电平的时间各占一半)，所以逻辑门电路的平均功率损耗为

$$\boldsymbol{P_D = V_{CC}\left(\frac{I_{CCH} + I_{CCL}}{2}\right)} \tag{3.2}$$

CMOS 门电路和双极型门电路系列相比较，它们的功率损耗非常小。然而，CMOS 门电路的功率损耗和它的工作频率有关。频率为零时，典型的静态功率损耗在微瓦(μW)/门的范围，而在最大工作频率下，它的功率损耗可以降低到毫瓦(mW)/门的范围；因此，功率损耗通常是在给出的特定频率下的值。例如对于 HC 系列，静态的功率损耗为 2.75 μW/门，而在 1 MHz 的工作频率下的功率损耗为 600 μW/门。

双极型门电路的功率损耗和频率无关。例如，ALS 系列的功率损耗和频率无关，其功率损耗为 1.4 mW/门，F 系列的功率损耗为 6 mW/门。

输入/输出逻辑电平 V_{IL}为一个逻辑门的输入电压的低电平值，V_{IH}为输入电压的高电平值。供电电压为5 V的CMOS可以接受的最大的电压V_{IL}值是1.5 V，最小的电压V_{IH}值是3.5 V。双极型逻辑门可以接受的最大的电压V_{IL}值是0.8 V，最小的电压V_{IH}值是2 V。

V_{OL}为一个逻辑门的输出电压的低电平值，V_{OH}为输出电压的高电平值。供电电压为5 V的CMOS可以接受的最大的电压V_{OL}值是0.33 V，最小的电压V_{OH}值是4.4 V。双极型逻辑门可以接受的最大的电压V_{OL}值是0.4 V，最小的电压V_{OH}值是2.4 V。所有的值如数据表给出的是在某个工作状态下的值。

速度-功率乘积(SPP) 在考虑延迟时间和功率损耗的情况下，这个参数可以用来测量逻辑电路的运行状况。它在比较CMOS和双极型系列(或称CMOS门和TTL门)之间各种逻辑门系列的性能时特别有用。

逻辑电路的SPP是延迟时间和功率损耗的乘积，计算值的单位是焦耳(J)，这是一个能量的单位。公式如下：

$$\mathbf{SPP} = t_p P_D \tag{3.3}$$

例3.23 某个门电路有5 ns的延迟时间，在直流电源电压为5 V时，它的$I_{CCH}=1$ mA，$I_{CCL}=2.5$ mA。求出速度-功率乘积。

解：

$$P_D = V_{CC}\left(\frac{I_{CCH}+I_{CCL}}{2}\right) = 5\text{ V}\left(\frac{1\text{ mA}+2.5\text{ mA}}{2}\right) = 5\text{ V}(1.75\text{ mA}) = 8.75\text{ mW}$$

$$\text{SPP} = (5\text{ ns})(8.75\text{ mW}) = \mathbf{43.75\text{ pJ}}$$

相关问题： 如果电路的延迟时间为15 ns，SPP为150 pJ，那么它的平均功率损耗是多少？

扇出和负载 一个逻辑门的扇出是指它的输出可以连接相同IC系列的输入端的最多数目，同时输出电压电平仍然保持在规定的范围以内。由于电路技术的原因，扇出仅仅对于双极型逻辑是一个重要参数。因为CMOS电路的阻抗很高，所以扇出数很大，但是电容效应会影响它的扇出数。

◇ 一个门的扇出数在很大程度上意味着它可以连接更多的门的输入。

扇出定义为单位负载数。一个逻辑门的单位负载等于同样电路的一个输入。例如，一个74LS00与非门的单位负载等于另一个74LS系列(不一定是与非门)的一个输入。由于流出一个74LS00门电路输入的低电平电流为0.4 mA(I_{IL})，而输出可以接受的低电平电流为8.0 mA(I_{OL})，因此一个74LS00门电路在低电平状态下可以驱动的单位负载数是

$$\text{单位负载数} = \frac{I_{OL}}{I_{IL}} = \frac{8.0\text{ mA}}{0.4\text{ mA}} = 20$$

图3.53给出了LS逻辑门驱动的相同电路技术的其他门的数目，这里驱动门的数目取决于特定的电路技术。例如，双极型74LS系列门的最大可驱动门的输入数目(单位负载数)是20。

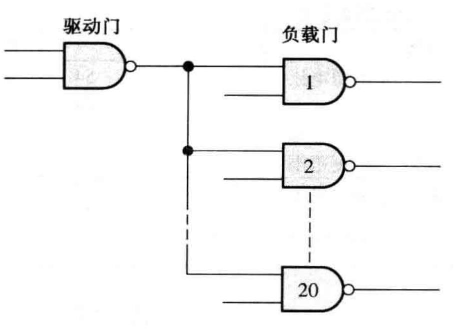


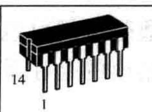
图3.53 LS系列与非门输出的最大扇出数是最大20个LS系列门的输入

3.7.4 数据表

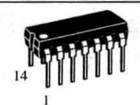
一个典型的数据表由一张资料页组成，其中给出逻辑框图和封装情况，建议的工作条件，电气和开关特性，还有一些其他的信息。图3.54和图3.55分别给出了74LS00和74HC00A的部分数据表。数据表的大小各不相同，有些数据表的内容很多。

四 2 输入与非门
- ESD > 3500 V

SN54/74LS00	四 2输入与非门 低功耗肖特基



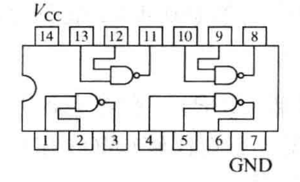
J后缀
陶瓷封装
例图632-08



N后缀
塑料封装
例图646-06

D后缀
SOIC
例图 751A-02

原始资料
SN54LSXXJ 陶瓷
SN74LSXXN 塑料
SN74LSXXD SOIC



SN54/74LS00

在工作温度范围以外的 DC(直流)特性(除非特别说明)

符号	参数		限制			单位	测试条件
			最小	典型	最大		
V_{IH}	输入高电平电压		2.0			V	所有输入不超过允许的高电平电压
V_{IL}	输入低电平电压	54			0.7	V	所有输入不超过允许的低电平电压
		74			0.8		
V_{IK}	输入钳位二极管电压			-0.65	-15	V	V_{CC} = MIN, I_{IN} = -18 mA
V_{OH}	输出高电平电压	54	2.5	3.5		V	V_{CC} = MIN, I_{OH} = MAX, V_{IN} = V_{IH}或 V_{IL}
		74	2.7	3.5		V	(每个真值表)
V_{OL}	输出低电平电压	54,74		0.25	0.4	V	I_{OL} =4.0 mA V_{CC} = V_{CC} MIN, V_{IN} = V_{IL}
		74		0.35	0.5	V	I_{OL} =8.0 mA 或 V_{IH}(每个真值表)
I_{IH}	输入高电平电流				20	μA	V_{CC} = MAX, V_{IN} =2.7 V
					0.1	mA	V_{CC} = MAX, V_{IN} =7.0 V
I_{IL}	输入低电平电流				-0.4	mA	V_{CC} = MAX, I_N =0.4 V
I_{OS}	短路电流(注 1)		-20		-100	mA	V_{CC} = MAX
I_{CC}	电源提供的电流 总电流,输出高电平				1.6	mA	V_{CC} = MAX
	总电流,输出低电平				4.4		

注 1:每次仅需一个输出短路,第二次也一样。

AC(交流)特性(T_A =25℃)

符号	参数	限制			单位	测试条件
		最小	典型	最大		
t_{PLH}	关断延迟,输入到输出		9.0	15	ns	V_{CC} =5.0 V
t_{PHL}	关断延迟,输入到输出		10	15	ns	C_L =15 pF

安全工作范围

符号	参数		最小	典型	最大	单位
V_{CC}	电源电压	54	4.5	5.0	5.5	V
		74	4.75	5.0	5.25	
T_A	工作环境 温度范围	54	-55	25	125	℃
		74	0	25	70	
I_{OH}	输出电流——高电平	54,74			-0.4	mA
I_{OL}	输出电流——低电平	54			4.0	mA
		74			8.0	

图 3.54 74LS00 四 2 输入与非门的部分数据表

74LS00 四 2 输入与非门　高速 CMOS 硅晶体管门

MC54/74HC00A 的输出引脚和 LS00 完全相同。芯片的输入与 CMOS 的输出兼容；连接上拉电阻后，与 LSTTL 的输出兼容。

- 输出驱动能力：10 个 LSTTL 负载
- 输出可以直接与 CMOS、NMOS 和 TTL 连接
- 工作电压范围：2 ~ 6 V
- 低电平输出电流：1 μA
- CMOS 芯片的高抗噪声特性
- 与所需的标准编号 7A 的 JEDEC 兼容
- 芯片的组成结构：32 个 FET 或 8 个等效的门

MC54/74HC00A

J前缀
陶瓷封装
例图632-08

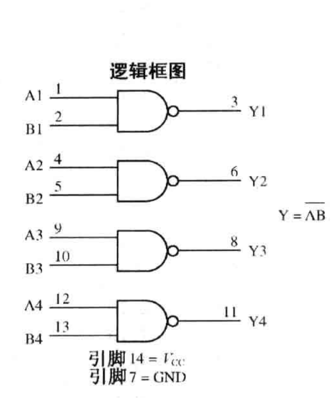
N前缀
塑料封装
例图646-06

D前缀
SOIC封装
例图751A-03

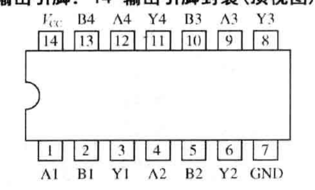
D前缀
TSSOP封装
例图948G-01

原始资料

MC54HCXXAJ	陶瓷
MC74HCXXAN	塑料
MC74HCXXAD	SOIC
MC74HCXXADT	TSSOP



输出引脚：14-输出引脚封装(顶视图)



功能表

输入		输出
A	B	Y
L	L	H
L	H	H
H	L	H
H	H	L

最大范围*

符号	参数	值	单位
V_{CC}	DC 电源电压(对地电压)	−0.5 ~ +7.0	V
V_{in}	DC 输入电压(对地电压)	−0.5 ~ V_{CC} +0.5	V
V_{out}	DC 输出电压(对地电压)	−0.5 ~ V_{CC} +0.5	V
I_{in}	DC 输入电流，每个引脚	±20	mA
I_{out}	DC 输出电流，每个引脚	±25	mA
I_{CC}	DC 电源电流，引脚 V_{CC} 和 GND	±50	mA
P_D	空气中的功率损耗，塑料或陶瓷、SOIC、TSSOP⁺	750 500 450	mW
T_{stg}	保存温度	−65 ~ +150	℃
T_L	导线温度，10 秒情况下 1 mm 塑料 DIP 封装，SOIC 或 TSSOP 封装 陶瓷 DIP 封装	260 300	℃

* 最大额定值是远低于芯片会损坏情况下的值。运行的功能指严格限制在推荐的工作条件下的功能。

⁺ 塑料 DIP 封装：−10 mW/℃，从 65°到 125℃。
陶瓷 DIP 封装：−10 mW/℃，从 100°到 125℃。
SOIC 封装：−7 mW/℃，从 65°到 125℃。
TSSOP 封装：−6.1 mW/℃，从 65°到 125℃。

图 3.55　74HC00A 四 2 输入与非门的部分数据表

建议的工作环境

符号	参数		最小	最大	单位
V_{CC}	DC 电源电压(对地电压)		2.0	6.0	V
V_{in}, V_{out}	DC 输入电压,输出电压(对地电压)		0	V_{CC}	V
T_A	工作温度,各种封装类型		−55	+125	℃
t_r, t_f	输入升降时间	$V_{CC}=2.0$ V	0	1000	ns
		$V_{CC}=4.5$ V	0	500	
		$V_{CC}=6.0$ V	0	400	

DC 特性(对地电压) MC54/74HC00A

符号	参数	工作条件	V_{CC} V	限制条件下的保证值 −55~25℃	≤85℃	≤125℃	单位
V_{IH}	最小高电平输入电压	$V_{out}=0.1$ V 或 $V_{CC}-0.1$ V $\lvert I_{out}\rvert \le 20$ μA	2.0	1.50	1.50	1.50	V
			3.0	2.10	2.10	2.10	
			4.5	3.15	3.15	3.15	
			6.0	4.20	4.20	4.20	
V_{IL}	最大低电平输入电压	$V_{out}=0.1$ V 或 $V_{CC}-0.1$ V $\lvert I_{out}\rvert \le 20$ μA	2.0	0.50	0.50	0.50	V
			3.0	0.90	0.90	0.90	
			4.5	1.35	1.35	1.35	
			6.0	1.80	1.80	1.80	
V_{OH}	最大高电平输出电压	$V_{in}=V_{IH}$或V_{IL} $\lvert I_{out}\rvert \le 20$ μA	2.0	1.9	1.9	1.9	V
			4.5	4.4	4.4	4.4	
			6.0	5.9	5.9	5.9	
		$V_{in}=V_{IH}$或V_{IL} $\lvert I_{out}\rvert \le 2.4$ mA	3.0	2.48	2.34	2.20	
		$\lvert I_{out}\rvert \le 4.0$ mA	4.5	3.98	3.84	3.70	
		$\lvert I_{out}\rvert \le 5.2$ mA	6.0	5.48	5.34	5.20	
V_{OL}	最大低电平输出电压	$V_{in}=V_{IH}$或V_{IL} $\lvert I_{out}\rvert \le 20$ μA	2.0	0.1	0.1	0.1	V
			4.5	0.1	0.1	0.1	
			6.0	0.1	0.1	0.1	
		$V_{in}=V_{IH}$或V_{IL} $\lvert I_{out}\rvert \le 2.4$ mA	3.0	0.26	0.33	0.40	
		$\lvert I_{out}\rvert \le 4.0$ mA	4.5	0.26	0.33	0.40	
		$\lvert I_{out}\rvert \le 5.2$ mA	6.0	0.26	0.33	0.40	
I_{in}	最大输入漏电流	$V_{in}=V_{CC}$或GND	6.0	±0.1	±0.1	±0.1	μA
I_{CC}	最大静态工作电流(单个芯片封装)	$V_{in}=V_{CC}$或GND $I_{out}=0$ μA	6.0	1.0	10	40	μA

AC 特性($C_L=50$ pF 输入 $t_r=t_f=6$ ns)

符号	参数	V_{CC} V	限制条件下的保证值 −55~25℃	≤85℃	≤125℃	单位
t_{PLH} t_{PHL}	最大传输延迟,输入 A 或 B 到输出 Y	2.0	75	95	110	ns
		3.0	30	40	55	
		4.5	15	19	22	
		6.0	13	16	19	
t_{TLH} t_{THL}	最大输出电容	2.0	75	95	110	ns
		3.0	27	32	36	
		4.5	15	19	22	
		6.0	13	16	19	
C_{in}	最大输入电容		10	10	10	pF

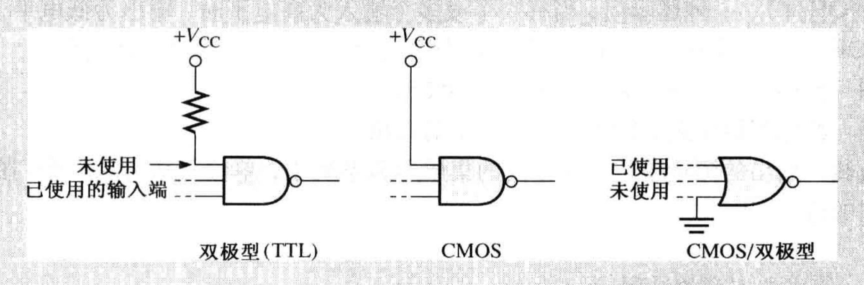
符号	参数	典型值:@25℃, $V_{CC}=5.0$ V, $V_{EE}=0$ V	单位
C_{PD}	功率损耗电容(每个缓冲区)	22	pF

图 3.55(续) 74HC00A 四 2 输入与非门的部分数据表



应用提示

双极型(TTL)和 CMOS 的未使用输入端应该连接到恰当的逻辑电平上(高电平或低电平)。对于与门/与非门,建议未使用的输入端连接到 V_{CC}(双极型则通过一个 1 kΩ 的电阻连接到 V_{CC}),对于或门/或非门,未使用的输入端应该连接到地。





集成电路技术

在本书的第 12 章将会介绍这些技术及其他的一些注意事项。

3.7 节　温故而知新

1. 列出两种类型的广泛使用的集成电路(IC)技术。
2. 判断下列 IC 的定义:
   (a)LS　(b)ALS　(c)F　(d)HC　(e)AC　(f)HCT　(g)LV
3. 按照逻辑功能来判断下列芯片:
   (a)74LS04　(b)74HC00　(c)74ALS10
   (d)7432　(e)74ACT11　(f)74AHC02
4. 哪一种 IC 技术具有最低的功率损耗?
5. 术语十六反相器指的是什么? 四 2 输入与非门指的是什么?
6. 一个正脉冲加在一个反相器的输入端。输入的上升沿到输出的上升沿的时间是10 ns。输入的下降沿到输出的下降沿的时间是 8 ns。那么 t_{PLH} 和 t_{PHL} 的值是多少?
7. 某个门的传输延迟时间为 6 ns,功率损耗为 3 mW。计算出它的速度-功率乘积。
8. 定义 I_{CCL} 和 I_{CCH}。
9. 定义 V_{IL} 和 V_{IH}。
10. 定义 V_{OL} 和 V_{OH}。

关键词

与门(AND)　一种逻辑门,只有在所有的输入都为高电平时输出才为高电平。

双极型　一种集成逻辑电路的类型,产品有双极型晶体管,通常为 TTL。

布尔代数　逻辑电路的数学基础。

CMOS　互补金属氧化物半导体;集成逻辑电路的一种,常见的应用有场效应管。

求反　一个数的反码或反相。低电平产生高电平,0 变为 1。

同或门 一种逻辑门，当两个输入为相反电平时，输出为低电平。

异或门(XOR) 一种逻辑门，当两个输入为相反电平时，输出为高电平。

扇出 一个逻辑门所能驱动的相同系列的等效门输入的数目。

反相器 一种把它的输入反相的逻辑电路。

与非(NAND)门 一种逻辑门，当所有输入为高电平时，输出为低电平。

或非(NOR)门 一种逻辑门，当有一个或多个输入为高电平时，输出为低电平。

或门(OR) 一种逻辑门，当有一个或多个输入为高电平时，输出为高电平。

时序图 给出所有波形的对应时间关系的波形图。

真值表 给出逻辑电路的输入和相应输出的表格。

单位负载 扇出的测量单位。在相同的集成电路系列中，单位负载表示一个门的输出所对应的一个门的输入。

判断题 (答案在本章的结尾。)

1. 反相器完成或非的操作。
2. 与门仅有两个输入。
3. 如果或门的任意一个输入为1，那么它的输出为1。
4. 如果与门的所有输入为1，那么它的输出为0。
5. 与非门的输出和一个与门的输出相反。
6. 或非门可以认为是一个或门紧接着连接一个非门。
7. 如果一个异或门的输入不同，那么输出为0。
8. 固定逻辑功能集成电路的两种类型为双极型和NMOS。
9. BCD表示二-十进制编码。
10. 扇出指给出的可以驱动的相同门的数目。

自测题 (答案在本章的结尾。)

1. 当反相器的输入为高电平1时，输出是
   (a)高电平或者1 (b)低电平或者1 (c)高电平或者0 (d)低电平或者0
2. 反相器执行的运算称为
   (a)求反码 (b)确定 (c)反相 (d)答案(a)和(c)
3. 与门的输入为A、B、C，输出何时为1(高电平)?
   (a)$A=1$, $B=1$, $C=1$ (b)$A=1$, $B=0$, $C=1$ (c)$A=0$, $B=0$, $C=0$
4. 或门的输入为A、B、C，输出何时为1(高电平)?
   (a)$A=1$, $B=1$, $C=1$ (b)$A=0$, $B=0$, $C=1$
   (c)$A=0$, $B=0$, $C=0$ (d)答案(a)、(b)和(c)
   (e)只有答案(a)和(b)
5. 一个脉冲加在一个2输入与非门的每一个输入上。某一个脉冲在$t=0$为高电平，而在$t=1$时变为低电平。另一个脉冲在$t=0.8$ ms时变为高电平，而在$t=3$ ms时变为低电平。输出脉冲应当描述为
   (a)在$t=0$时变为低电平，而在$t=3$ ms时变为高电平。
   (b)在$t=0.8$ ms时变为低电平，而在$t=3$ ms时变为高电平。
   (c)在$t=0.8$ ms时变为低电平，而在$t=1$ ms时变为高电平。
   (d)在$t=0.8$ ms时变为低电平，而在$t=1$ ms时变为低电平。

6. 一个脉冲加在一个2输入或非门的每一个输入上。某一个脉冲在$t=0$时为高电平，而在$t=1$ ms时变为低电平。另一个脉冲在$t=0.8$ ms时变为高电平，而在$t=3$ ms时变为低电平。输出脉冲应当描述为
   (a)在$t=0$时变为低电平，而在$t=3$ ms时变为高电平。
   (b)在$t=0.8$ ms时变为低电平，而在$t=3$ ms时变为高电平。
   (c)在$t=0.8$ ms时变为低电平，而在$t=1$ ms时变为高电平。
   (d)在$t=0.8$ ms时变为高电平，而在$t=1$ ms时变为低电平。
7. 一个脉冲加在一个异或门的每一个输入上。某一个脉冲在$t=0$时变为高电平，而在$t=1$ ms时变为低电平。另一个脉冲在$t=0.8$ ms时变为高电平，而在$t=3$ ms时变为低电平。输出脉冲应当描述为
   (a)在$t=0$时变为高电平，而在$t=3$ ms时变为低电平。
   (b)在$t=0$时变为高电平，而在$t=0.8$ ms时变为低电平。
   (c)在$t=1$ ms时变为高电平，而在$t=3$ ms时变为低电平。
   (d)答案(b)和(c)
8. 一个正向脉冲应用于反相器。从输入前沿到输出前沿的时间间隔为7 ns。这个参数是
   (a)速度-功率乘积　(b)传播延迟时间t_{PHL}　(c)传播延迟时间t_{PLH}　(d)脉冲宽度
9. 当测量一个脉冲波形时，频率由如下哪一项确定：
   (a)使用另一个设置　(b)测量占空比　(c)求出周期的倒数　(d)使用另一种类型的仪器

习题 （奇数题的答案在本书的结尾。）

3.1节　反相器

1. 如图3.56所示的输入波形加在一个反相器上。画出与输入波形相关的输出波形的时序图。

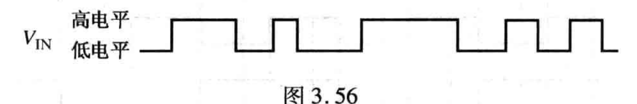


图3.56

2. 反相器级联网络如图3.57所示。如果高电平加在输入点A，确定点B到F所对应的输出波形。

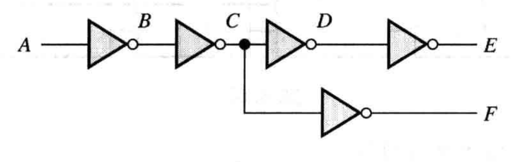


图3.57

3. 如果图3.56的波形加在图3.57的A点，确定点B到F所对应的输出波形。

3.2节　与门

4. 画出一个4输入与门的矩形轮廓符号。
5. 输入波形如图3.58所示，确定2输入与门的输出X，使用时序图画出与输入对应的正确的输入/输出关系。
6. 如图3.59所示，重复习题5，画出输出波形。

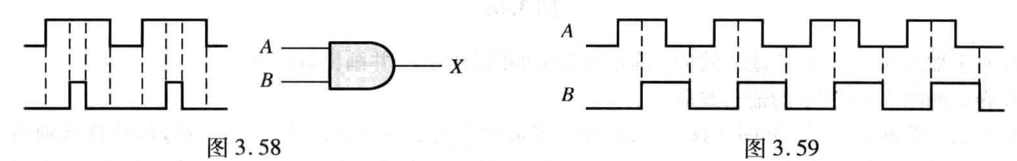


图3.58　　图3.59

7. 加在一个 3 输入与门的输入波形如图 3.60 所示。利用时序图画出与输入对应的输出波形。

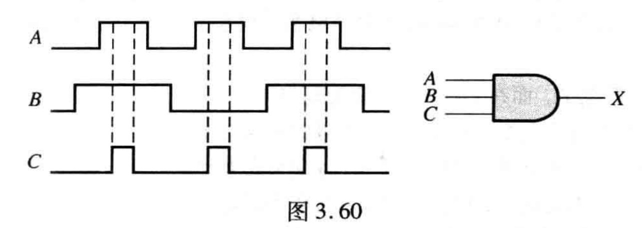


图 3.60

8. 加在一个 4 输入与门的输入波形如图 3.61 所示。利用时序图画出与输入对应的输出波形。

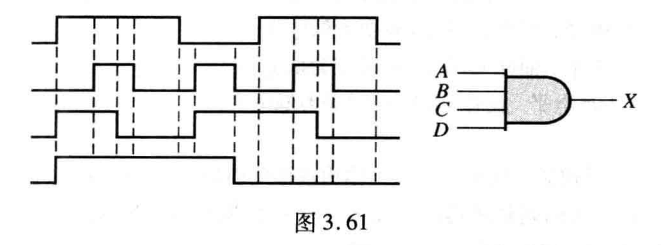


图 3.61

3.3 节　或门

9. 当输入波形如图 3.59 所示时，确定一个 2 输入或门的输出，并且绘制出时序图。

10. 对 3 输入或门重复习题 7。

11. 对 4 输入或门重复习题 8。

12. 对如图 3.62 所示的 5 个输入波形，确定 5 输入与门的输出，以及 5 输入或门的输出。画出时序图。

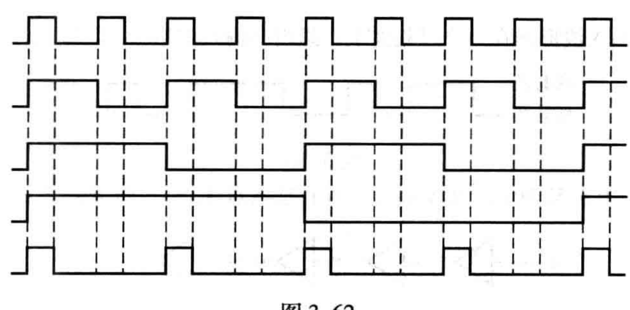


图 3.62

13. 画出一个 4 输入或门的矩形轮廓符号。

14. 给出一个 3 输入或门的真值表。

3.4 节　与非门

15. 对于如图 3.63 所示的输入波形集合，确定所给出的门的输出，并画出时序图。

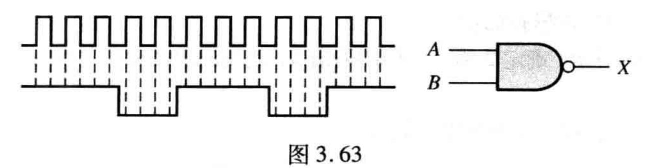


图 3.63

16. 对于如图 3.64 所示的输入波形，确定所给出的门的输出，并画出时序图。

17. 确定如图 3.65 所示的输出波形。

18. 正如所学到的那样，如图 3.66 所示的两个逻辑符号表示等价的运算。两种符号的严格区别在于功能。对于与非门符号来说，两个输入上的高电平给出一个低电平。对于非-或符号来说，输入只要有

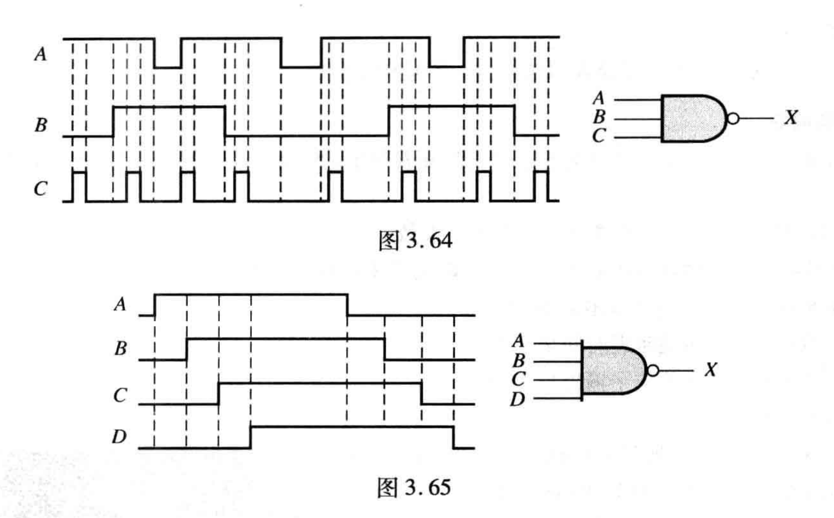


图 3.64

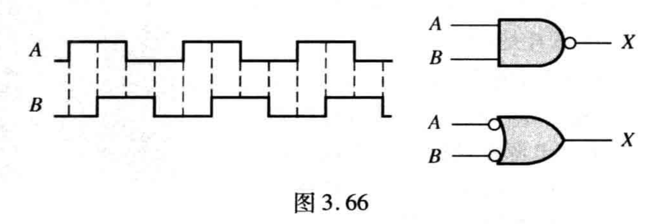


图 3.65

一个低电平就给出一个高电平输出。利用这两种功能，为给定的输入画出每个门的输出，这两个输出是相同的。

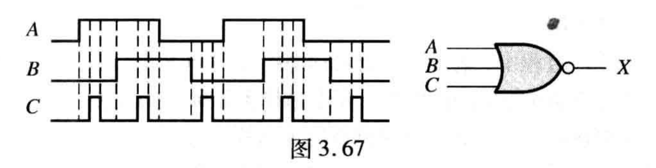


图 3.66

3.5 节　或非门

19. 为一个 2 输入或非门重复习题 15。
20. 确定图 3.67 中的输出波形，并绘制时序图。

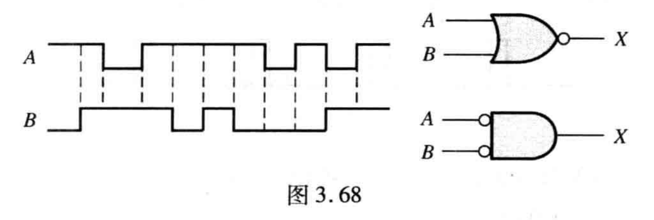


图 3.67

21. 为一个 4 输入或非门重复习题 17。
22. 或非和非-与符号表示等价运算，但是它们在功能上是不同的。对于或非符号来说，输入只要有一个高电平，就给出低电平输出。对于非-与符号来说，输入上有两个低电平才给出一个高电平输出。利用这两种功能，在图 3.68 中对于两个门所给定的输入画出输出波形，这两个输出是相同的。

图 3.68

3.6 节　异或门和同或门

23. 异或门和或门在逻辑运算上是怎样区分的?
24. 为异或门重复习题 15。

25. 为同或门重复习题15。

26. 为图3.59所示的输入确定异或门的输出，并画出时序图。

3.7节 固定功能逻辑

27. 在比较某些逻辑器件时，注意对于某个类型随着频率的增加，功率损耗也增加。这是指双极型还是CMOS?

28. 使用图3.54和图3.55的数据表，确定如下参数：

(a)74LS00在最大供电电压和占空比50%时的功率损耗是多少?

(b)74LS00的最大高电平输出是多少?

(c)74LS00的最大传输延迟时间是多少?

(d)74HC00A的最大低电平输出是多少?

(e)74HC00A的最大传输延迟时间是多少?

29. 根据图3.69中示波器所显示的确定 t_{PLH} 和 t_{PHL}。每个通道的读数指示为电压每刻度和秒每刻度。

30. 门A的 $t_{PLH} = t_{PHL} = 6$ ns。门B的 $t_{PLH} = t_{PHL} = 10$ ns。哪一个门的工作频率高?

31. 如果一个逻辑门工作在供电电压+5 V和平均拉电流4 mA时，那么它的功率损耗是多少?

32. 变量 I_{CCH} 表示一个集成芯片的所有输出为高电平时电源提供的直流电流。变量 I_{CCL} 表示一个集成电路芯片的所有输出为低电平时电源提供的直流电流。对于74LS00集成电路芯片，确定其在所有四个门的输出为高电平时典型的功率损耗(参照图3.54)。

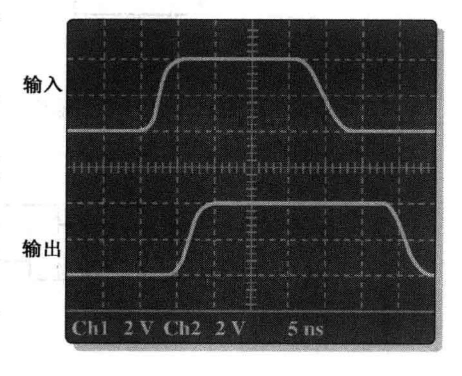


图3.69

特殊设计问题

33. 传感器用来检测存储在一个液体桶里的化学溶液的压力和温度。当超过某个设定值时，对应的那个传感器电路会产生一个高电平电压。当压力或温度太大时，必须输入一个低电平电压以驱动报警装置。设计一个这样的电路。

34. 在一个自动装置的生产过程中，电子元件被自动地插入到PCB上。在插入工具启动前，PCB必须准确定位，因为元件必须插在指定的位置里。每个这样预先确定的条件由一个高电平电压指定。插入工具由一个低电平电压启动。设计一个电路以完成这样的过程。

35. 修改图3.16中的频率计数器，产生一个低电平有效的使能脉冲而不是在1 ms间隔时的高电平。

36. 假设图3.16中的使能信号的波形如图3.70所示，波形B仍然有效。设计一个电路，只有在使能信号为低电平时，才产生一个高电平有效的复位脉冲。

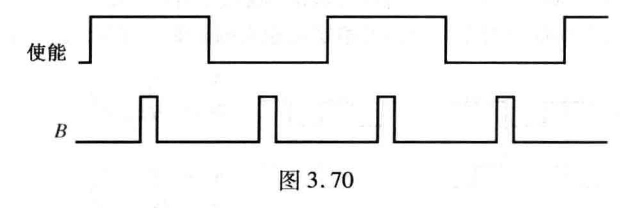


图3.70

37. 设计一个电路满足图3.71中框图的要求，电路会使一辆汽车在点火开关关闭15秒以后自动熄灭前灯。假设需要一个低电平把灯熄灭。

38. 修改图3.25中的入侵报警逻辑电路，使得两个附加的有两扇窗和一个门的房间能够得到防护。

39. 进一步修改习题38的电路，更改输入传感器的Open(打开)=LOW(低电平)和Closed(关闭)=HIGH(高电平)。

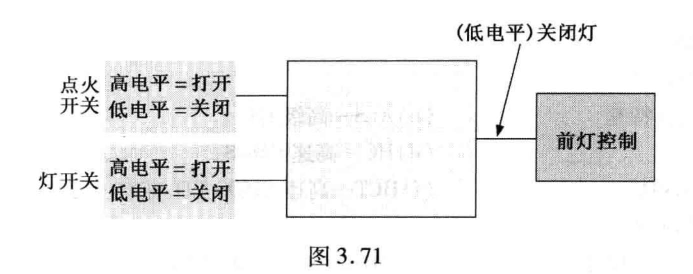


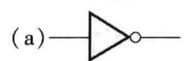
图3.71

答案

温故而知新

3.1节　反相器

1. 反相器的输入为1时，输出为0。

(a) 

(b) 一个输出端的负脉冲(高到低然后低到高)。

3.2节　与门

1. 只有当与门的所有输入为高电平时，它的输出才为高电平。
2. 当与门的一个或多个输入为低电平时，它的输出为低电平。
3. 5输入与门：当 $ABCDE = 11111$ 时 $X = 1$，所有其他的 $ABCDE$ 组合都使得 $X = 0$。

3.3节　或门

1. 当或门的一个或多个输入为高电平时，它的输出为高电平。
2. 只有当或门的所有输入为低电平时，它的输出才为低电平。
3. 3输入或门：当 $ABC = 000$ 时 $X = 0$，所有其他的 ABC 组合都使得 $X = 1$。

3.4节　与非门

1. 只有当与非门的所有输入为高电平时，它的输出才为低电平。
2. 当与非门的一个或多个输入为低电平时，它的输出为高电平。
3. 与非门：所有的输入为高电平时，输出低电平称为低电平有效输出；非-或运算：当一个或多个输入为低电平时，输出高电平称为高电平有效输出。这两种运算在真值表中是一样的。
4. $X = \overline{ABC}$

3.5节　或非门

1. 只有当或非门的所有输入为低电平时，它的输出才为高电平。
2. 当或非门的一个或多个输入为高电平时，它的输出为低电平。
3. 或非门：当一个或多个输入为高电平时，输出低电平称为低电平有效输出。非-与运算：当所有输入为低电平时，输出高电平称为高电平有效输出。这两种运算在真值表中是一样的。
4. $X = \overline{A + B + C}$

3.6节　异或门和同或门

1. 当异或门的两个输入是相反电平时，它的输出为高电平。
2. 当同或门的两个输入为相同电平时，它的输出为高电平。
3. 数位加到异或门的输入：输出为高电平时，两个数位的电平不同。

3.7 节　固定功能逻辑

1. CMOS 和双极型(TTL)

2. (a) LS—低功耗肖特基　　(b) ALS—高级 LS
   (c) F—高速 TTL　　(d) HC—高速 CMOS
   (e) AC—高级 CMOS　　(f) HCT—高速 CMOS TTL 兼容
   (g) LV—低电压 CMOS

3. (a) 74LS04—十六反相器　　(b) 74HC00—四 2 输入与非门
   (c) 74ALS10—三 3 输入与非门　　(d) 7432—四 2 输入或门
   (e) 74ACT11—三 3 输入与门　　(f) 74AHC02—四 2 输入或非门

4. 超低功耗—CMOS。

5. 十六反相器封装；四 2 输入与非门封装。

6. t_{PLH} = 10 ns；t_{PHL} = 8 ns。

7. 18 pJ

8. I_{CCL}— 直流低电平输出状态下提供的电流；I_{CCH}— 直流高电平输出状态下提供的电流。

9. V_{IL}— 低电平输入电压；V_{IH} - 高电平输入电压。

10. V_{OL}— 低电平输出电压；V_{OH} - 高电平输出电压。

例题的相关问题

3.1　时序图不受影响。

3.2　参照表 3.17。

表 3.17

输入 ABCD	输出 X	输入 ABCD	输出 X
0000	0	1000	0
0001	0	1001	0
0010	0	1010	0
0011	0	1011	0
0100	0	1100	0
0101	0	1101	0
0110	0	1110	0
0111	0	1111	1

3.3　参照图 3.72。

3.4　输出波形和输入 A 相同。

3.5　参照图 3.73。

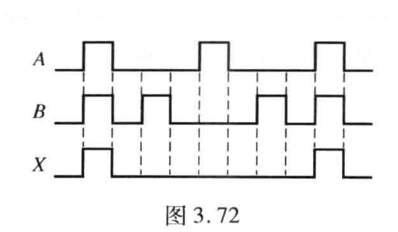


图 3.72

A
B
C
X
C = 高电平

图 3.73

3.6　和例题的结果相同。

3.7　参照图 3.74。

3.8　参照图 3.75。

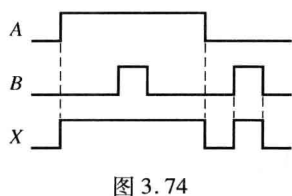


图 3.74

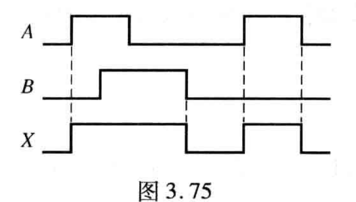


图 3.75

3.9　参照图 3.76。

3.10　参照图 3.77。

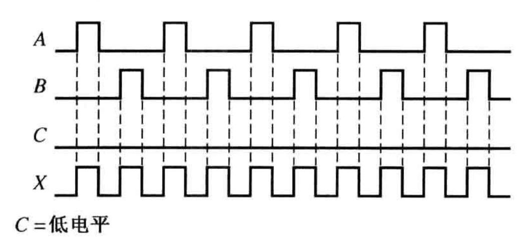




图 3.76

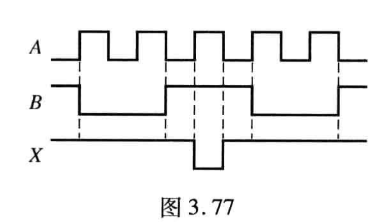


图 3.77

3.11　参照图 3.78。

3.12　使用一个 3 输入与非门。

3.13　使用一个 4 输入与非门作为非-或运算。

3.14　参照图 3.79。

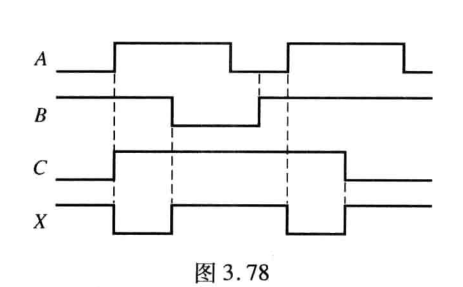


图 3.78

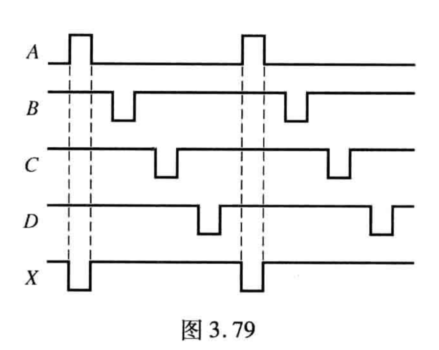


图 3.79

3.15　参照图 3.80。

3.16　参照图 3.81。

3.17　使用一个 2 输入或非门。

3.18　一个 3 输入与非门。

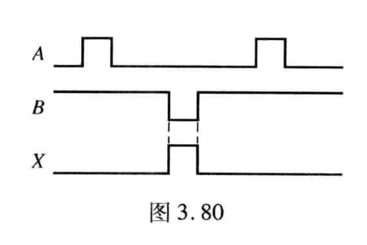


图 3.80

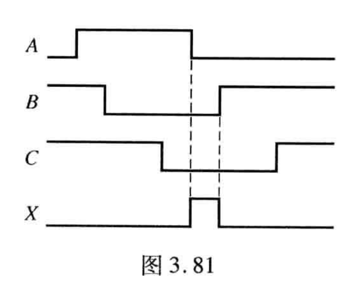


图 3.81

3.19 输出总是低电平。输出是一条直线。

3.20 如果两个电路产生相同的输出，异或门将不会同时检测故障。

3.21 输出不受影响。

3.22 在最高频率下，门电路具有 4 ns 的 t_{PLH} 和 t_{PHL}。

3.23 10 mW

判断题

1.F 2.F 3.T 4.F 5.T 6.T 7.F 8.F 9.T 10.T

自测题

1.(d) 2.(d) 3.(a) 4.(e) 5.(c)
6.(a) 7.(d) 8.(b) 9.(c)

第4章 布尔代数和逻辑化简

章节提纲

4.1 布尔运算和表达式

布尔代数中使用的术语为变量、反码和文字。变量用来表示一个作用、状态或数据(通常是斜体大写字母或单词)。任何一个单变量可以具有1或者0的值。反码是变量的反相，并且由变量上方的横杠(上划线)表示。例如，变量A的反码是$\overline{A}$，如果$A=1$，那么$\overline{A}=0$。如果$A=0$，那么$\overline{A}=1$。变量A的反码读做"非A"或者"A杠"。有时候用撇符号而不是用上划线来表示变量的反码；例如，B'就表示B的反码。在本书中，仅使用上划线。一个文字变量就是一个变量或者变量的反码。



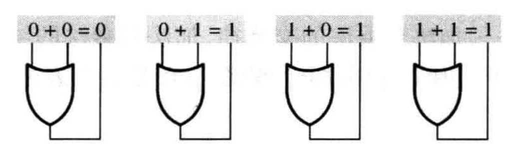
计算机小知识

在微处理器中，算术逻辑单元(ALU)根据程序的指令，对数字数据执行算术和布尔逻辑运算。逻辑运算等价于熟知的基本逻辑门的运算，但是每次至少处理8位。布尔逻辑指令的例子是与、或、非和异或，它们被称为助记符。汇编语言程序使用助记符来指定运算。另一个称为汇编器的程序把助记符翻译成可以被微处理器理解的二进制代码。

4.1.1 布尔加法

◇ 或门就是一个布尔加法的表示形式。

回顾第3章，布尔加法等价于或运算，其基本法则用或门表示如下：



在布尔代数中，和项就是文字变量的相加。在逻辑电路中，和项由或运算产生，其中没有与运算。和项的一些例子为$A+B$、$A+\overline{B}$、$A+B+\overline{C}$和$\overline{A}+B+C+\overline{D}$。

和项中有一个或者多个文字变量为1时，和项就等于1。只有当每个文字变量都是0时，和项才等于0。

例 4.1 已知 $A+\overline{B}+C+\overline{D}=0$，求出 A、B、C 和 D 的值。

解： 要使得和项等于0，那么和项中的每一个文字变量必须为0。因此 $A=\mathbf{0}$，$\overline{B}=0$ 那么 $B=\mathbf{1}$，$C=\mathbf{0}$，$\overline{D}=0$ 那么 $D=\mathbf{1}$。

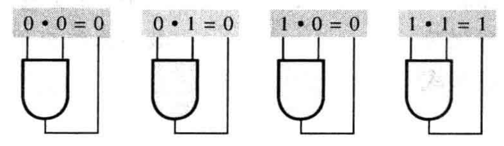
$$A+\overline{B}+C+\overline{D}=0+\overline{1}+0+\overline{1}=0+0+0+0=0$$

相关问题[①]：已知 $\overline{A}+B=0$、求出 A、B 的值。

4.1.2 布尔乘法

◇ 与门就是一个布尔乘法的表示形式。

同样回顾第3章，布尔乘法等价于与门运算，其基本法则用与门表示如下：



在布尔代数中，乘积项就是字母变量的乘积。在逻辑电路中，乘积项由与门运算产生，没有或的运算。乘积项的一些例子为 AB、$A\overline{B}$、ABC 和 $A\overline{B}C\overline{D}$ 。

当一个乘积项的每一个文字变量都是1时，乘积项等于1。当一个或者多个文字变量为0时，乘积项就等于0。

例 4.2 确定使得乘积项 $A\overline{B}C\overline{D}$ 等于1的 A、B、C 和 D 的值。

解： 为了使得乘积项等于1，那么该项中的每一个文字变量都必须是1。所以，$A=\mathbf{1}$，$\overline{B}=1$ 那么 $B=\mathbf{0}$，$C=\mathbf{1}$，$\overline{D}=1$ 那么 $D=\mathbf{0}$。

$$A\overline{B}C\overline{D}=1\cdot\overline{0}\cdot1\cdot\overline{0}=1\cdot1\cdot1\cdot1=1$$

相关问题： 确定使得乘积项 $\overline{A}\overline{B}$ 等于1的 A 和 B 的值。

4.1 温故而知新 （答案在本章的结尾。）

1. 如果 $A=0$，那么 $\overline{A}$ 等于什么？
2. 确定 A、B 和 C 的值，使得和项 $\overline{A}+\overline{B}+C$ 等于0。
3. 确定 A、B 和 C 的值，使得乘积项 $A\overline{B}C$ 等于1。

4.2 布尔代数的定律和法则

4.2.1 布尔代数的定律

布尔代数的基本定律——加法和乘法的交换律、加法和乘法的结合律及分配律——和通常的代数的定律一样。每一个定律都会由两个或者三个变量来表述，但是变量的个数不局限于此。

交换律 两变量加法的交换律可以写做

$$A+B=B+A \tag{4.1}$$

---

① 答案在本章的结尾。

这个定律表明对变量进行或运算的次序不会对结果产生影响。记住，布尔代数应用于逻辑电路时，加法和或运算是一样的。图 4.1 说明了应用于或门的交换律，并且指出变量加在哪个输入上并不重要。(符号≡的意思是“恒等于”。)

图 4.1　加法交换律的应用

两变量乘法的交换律为

$$AB = BA \tag{4.2}$$

这个定律表明变量与运算的次序不会对结果产生影响。图 4.2 给出了这个定律应用在与门的情况。

图 4.2　乘法交换律的应用

结合律　3 个变量的加法结合律可以写做下面的形式：

$$A + (B + C) = (A + B) + C \tag{4.3}$$

这个定律表明在对多于两个的变量进行或运算时，无论把哪些变量分为一组，其结果相同。图 4.3 给出了这个定律应用于 2 输入或门的情况。

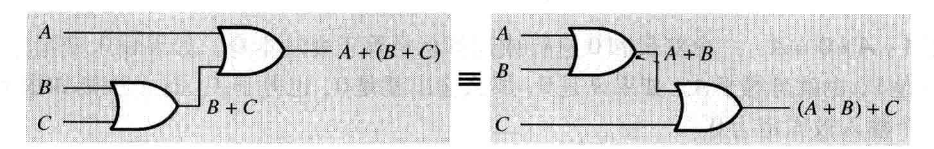


图 4.3　加法结合律的应用。打开文件 F04-03 检验该操作

3 变量的乘法的结合律可以写做：

$$A(BC) = (AB)C \tag{4.4}$$

这个定律表明当对多于两个的变量进行与运算时，变量被分组的次序不会对结果产生影响。图 4.4 给出了这个定律应用于 2 输入与门的情况。

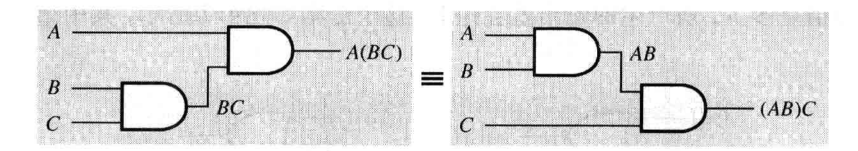
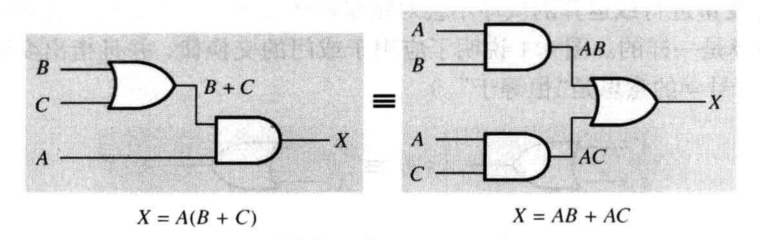


图 4.4　乘法交换律的应用。打开文件 F04-04 检验该操作

分配律　3 个变量的分配律可以写做：

$$A(B + C) = AB + AC \tag{4.5}$$

这个定律表明对两个或者更多的变量进行或运算，然后把结果和一个单变量相与，等价于把这个单变量和这两个或者更多变量中的每一个变量分别进行与运算，然后再把这些乘积进行或运算。分配律同样表示了因式分解的过程，共同变量 A 从乘积项中分解出来，例如，$AB + AC = A(B + C)$。图 4.5 以逻辑门的形式解释了分配律。



$X = A(B + C)$　　　$X = AB + AC$

图 4.5　分配律的应用。打开文件 F04-05 检验该操作

4.2.2　布尔代数的法则

表4.1 列出了 12 个基本法则，这在使用和化简布尔表达式中非常有用。法则 1 ~ 9 以它们在逻辑门中应用的方法进行介绍。法则 10 ~ 12 将会从先前已讨论的简单定律和法则中推导出来。

表 4.1　布尔代数的基本法则

1. $A + 0 = A$	**7.** $A \cdot A = A$
2. $A + 1 = 1$	**8.** $A \cdot \overline{A} = 0$
3. $A \cdot 0 = 0$	**9.** $\overline{\overline{A}} = A$
4. $A \cdot 1 = A$	**10.** $A + AB = A$
5. $A + A = A$	**11.** $A + \overline{A}B = A + B$
6. $A + \overline{A} = 1$	**12.** $(A + B)(A + C) = A + BC$

A、*B*、*C* 可以表示一个单变量或者变量的组合。

法则 1：$A + 0 = A$　一个变量和 0 进行或运算总是等于变量本身。如果输入变量是 1，输出变量 X 就是 1，也就是等于 A。如果 A 是 0，那么输出就是 0，也等于 A。这个法则如图 4.6所示，其中第二个输入被固定为 0。

$X = A + 0 = A$

图 4.6

法则 2：$A + 1 = 1$　一个变量和 1 进行或运算总是等于 1。或门的一个输入为 1，无论另一个输入的变量值是多少，所产生的输出就会是 1。这个法则如图 4.7 所示，其中第二个输入固定为 1。

$X = A + 1 = 1$

图 4.7

法则 3：$A \cdot 0 = 0$　一个变量和 0 进行与运算总是等于 0。任何时候与门的一个输入为 0，无论另一个输入的变量值是多少，输出就是 0。这个法则如图 4.8 所示，其中第二个输入固定为 0。

法则 4：$A \cdot 1 = A$　一个变量和 1 进行与运算总是等于变量本身。如果 A 为 0，与门的输出

就是 0；如果 A 为 1，与门的输出就是 1，因为现在这两个输入都是 1。这个法则如图 4.9 所示，其中第二个输入固定为 1。

$X = A \cdot 0 = 0$

图 4.8

$X = A \cdot 1 = A$

图 4.9

法则 5：$A + A = A$　一个变量和它本身进行或运算总是等于变量本身。如果 A 为 0，那么 $0 + 0 = 0$；如果 A 为 1，那么 $1 + 1 = 1$。这个法则如图 4.10 所示，其中两个输入都是相同的变量。

$X = A + A = A$

图 4.10

法则 6：$A + \overline{A} = 1$　一个变量和它的反码进行或运算总是等于 1。如果 A 为 0，那么 $0 + \overline{0} = 0 + 1 = 1$；如果 A 为 1，那么 $1 + \overline{1} = 1 + 0 = 1$。参见图 4.11，其中一个输入是另一个输入的反码。

$X = A + \overline{A} = 1$

图 4.11

法则 7：$A \cdot A = A$　一个变量和它本身进行与运算总是等于变量本身。如果 $A = 0$，那么 $0 \cdot 0 = 0$；如果 $A = 1$，那么 $1 \cdot 1 = 1$。图 4.12 解释了这个法则。

$X = A \cdot A = A$

图 4.12

法则 8：$A \cdot \overline{A} = 0$　一个变量和它的反码进行与运算总是等于 0。A 或者 $\overline{A}$ 总有一个为 0；当 0 应用于与门的输入时，输出就是 0。图 4.13 解释了这个法则。

法则 9：$\overline{\overline{A}} = A$　对一个变量进行两次求反后总是等于变量本身。如果开始于变量 A，对其进行一次求反（反相），得到 $\overline{A}$。如果再对 $\overline{A}$ 进行求反就会得到 A，也就是原始变量。这个法则如图 4.14 所示，图中使用了反相器。

$X = A \cdot \bar{A} = 0$

图 4.13

$\bar{\bar{A}} = A$

图 4.14

法则 10：$A + AB = A$ 这个法则可以使用分配律(法则 2 和法则 4)来证明，如下所示：

$$
\begin{aligned}
A + AB &= A(1 + B) && \text{因式分解(分配律)}\\
&= A \cdot 1 && \text{法则 2：}(1 + B) = 1\\
&= A && \text{法则 4：}A \cdot 1 = A
\end{aligned}
$$

这个证明如表 4.2 所示，其中给出了真值表和简化的逻辑电路。

表 4.2 法则 10：$A + AB = A$(打开文件 T04-02 检验该操作)

A	B	AB	$A + AB$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

法则 11：$A + \bar{A}B = A + B$ 这个法则证明如下：

$$
\begin{aligned}
A + \bar{A}B &= (A + AB) + \bar{A}B && \text{法则 10：}A = A + AB\\
&= (AA + AB) + \bar{A}B && \text{法则 7：}A = A \cdot A\\
&= AA + AB + A\bar{A} + \bar{A}B && \text{法则 8：加 }A\bar{A} = 0\\
&= (A + \bar{A})(A + B) && \text{因式分解}\\
&= 1 \cdot (A + B) && \text{法则 6：}A + \bar{A} = 1\\
&= A + B && \text{法则 4：舍去 1}
\end{aligned}
$$

这个证明如表 4.3 所示，其中给出了真值表和简化的逻辑电路。

表 4.3 法则 11：$A + \bar{A}B = A + B$(打开文件 T04-03 检验该操作)

A	B	$\bar{A}B$	$A + \bar{A}B$	$A + B$
0	0	0	0	0
0	1	1	1	1
1	0	0	1	1
1	1	0	1	1

法则 12：$(A+B)(A+C)=A+BC$ 这个法则的证明如下：

$$
\begin{aligned}
(A+B)(A+C) &= AA+AC+AB+BC && \text{分配律}\\
&= A+AC+AB+BC && \text{法则 7：} AA=A\\
&= A(1+C)+AB+BC && \text{因式分解（分配律）}\\
&= A\cdot 1+AB+BC && \text{法则 2：} 1+C=1\\
&= A(1+B)+BC && \text{因式分解（分配律）}\\
&= A\cdot 1+BC && \text{法则 2：} 1+B=1\\
&= A+BC && \text{法则 4：} A\cdot 1=A
\end{aligned}
$$

这个证明如表 4.4 所示，其中给出了真值表和简化的逻辑电路。

表 4.4　法则 12：$(A+B)(A+C)=A+BC$（打开文件 T04-04 检验该操作）

A	B	C	$A+B$	$A+C$	$(A+B)(A+C)$	BC	$A+BC$
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	1	1	1	1
1	0	0	1	1	1	0	1
1	0	1	1	1	1	0	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

4.2 节　温故而知新

1. 对表达式 $A+(B+C+D)$ 使用加法的结合律。
2. 对表达式 $A(B+C+D)$ 使用分配律。

4.3　狄摩根定理

◇ 使用狄摩根定理，把相与的变量上面的横杠分开，把与的符号变换成或的符号。

狄摩根（DeMorgan）第一定理如下所述：

变量乘积的反码等于变量反码的或。

另一种表述方式是

两个或者多个变量进行与运算之后的反码等于单个变量的反码再进行或运算。

对于两个变量，表述这个定理的公式为

$$\overline{XY}=\overline{X}+\overline{Y} \tag{4.6}$$

狄摩根第二定理表述如下：

变量之和的反码等于变量反码的乘积。

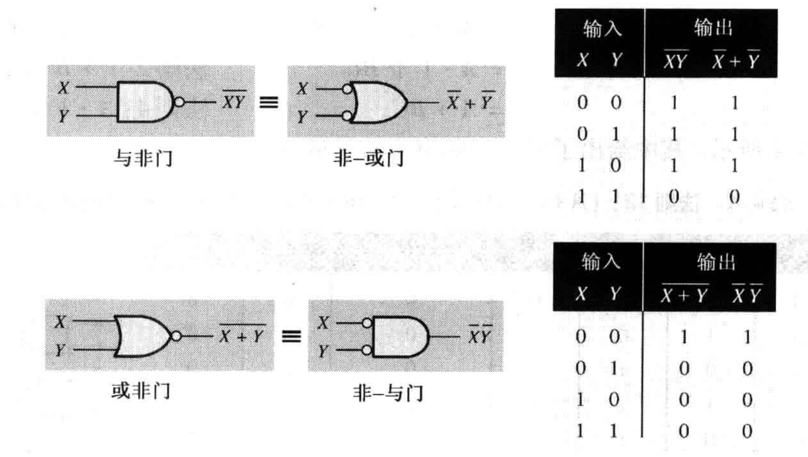
另一种表述方式是

对两个以上变量进行或运算之后的反码等于单个变量的反码再进行与运算的结果。

对于两个变量，表述这个定理的公式为

$$\overline{X+Y}=\overline{X}\,\overline{Y} \tag{4.7}$$

图4.15给出了式(4.6)和式(4.7)的真值表与等价的逻辑门。



输入		输出	
X	Y	$\overline{XY}$	$\overline{X}+\overline{Y}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

输入		输出	
X	Y	$\overline{X+Y}$	$\overline{X}\,\overline{Y}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

图4.15　用于解释狄摩根定理的等价门和相应的真值表。注意每个表中两个相同的输出列，这表示等价门执行相同的逻辑运算

如表述的那样，狄摩根定理也可应用于多于两个变量的表达式中。下面的例子就解释了狄摩根定理应用到3变量和4变量表达式的情况。

例4.3　把摩根定理应用于表达式 $\overline{XYZ}$ 和 $\overline{X+Y+Z}$。

解：

$$\overline{XYZ}=\overline{X}+\overline{Y}+\overline{Z}$$

$$\overline{X+Y+Z}=\overline{X}\,\overline{Y}\,\overline{Z}$$

相关问题：将狄摩根定理应用于表达式 $\overline{\overline{X}+\overline{Y}+\overline{Z}}$。

例4.4　将狄摩根定理应用于表达式 $\overline{WXYZ}$ 和 $\overline{W+X+Y+Z}$。

解：

$$\overline{WXYZ}=\overline{W}+\overline{X}+\overline{Y}+\overline{Z}$$

$$\overline{W+X+Y+Z}=\overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z}$$

相关问题：将狄摩根定理应用于表达式 $\overline{\overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z}}$。

如式(4.6)和式(4.7)所示，狄摩根定理中的每一个变量也可以表示其他变量的组合。例如，X 可以等于 $AB+C$ 项，而 Y 可以等于 $A+BC$ 项。所以可以把两个变量 $\overline{XY}=\overline{X}+\overline{Y}$ 的狄摩根定理由 $\overline{XY}=\overline{X}+\overline{Y}$ 表述，应用于表达式 $\overline{(AB+C)(A+BC)}$，就会得到以下结果：

$$\overline{(AB+C)(A+BC)}=(\overline{AB+C})+(\overline{A+BC})$$

注意前面的结果具有两个项：$\overline{AB+C}$ 和 $\overline{A+BC}$，那么可以对上述两项再次使用狄摩根定理 $\overline{X+Y}=\overline{X}\,\overline{Y}$，如下所示：

$$(\overline{AB+C})+(\overline{A+BC})=(\overline{AB})\overline{C}+\overline{A}(\overline{BC})$$

注意表达式中的两项 $\overline{AB}$ 和 $\overline{BC}$，可以再一次使用狄摩根定理。最后一次应用狄摩根定理给出了如下的结果：

$$(\overline{AB})\overline{C} + \overline{A}(\overline{BC}) = (\overline{A} + \overline{B})\overline{C} + \overline{A}(\overline{B} + \overline{C})$$

虽然可以使用布尔法则和定律进一步进行化简，但是已经不能再使用狄摩根定理了。

4.3.1　狄摩根定理的应用

下面的过程解释了狄摩根定理和布尔代数对于具体表达式的应用。

$$\overline{\overline{A + B\overline{C}} + D(\overline{E + \overline{F}})}$$

步骤 1：判断可以应用狄摩根定理的项，并将每一个项都看成单个变量。让 $\overline{A + B\overline{C}} = X$ 和 $D(\overline{E + \overline{F}}) = Y$。

步骤 2：因为 $\overline{X + Y} = \overline{X}\overline{Y}$，所以

$$\overline{(\overline{A + B\overline{C}}) + (D(\overline{E + \overline{F}}))} = (\overline{\overline{A + B\overline{C}}})(\overline{D(\overline{E + \overline{F}})})$$

步骤 3：使用法则 9（$\overline{\overline{A}} = A$）消去等式右边第一项上方的双重横杠（这并不是狄摩根定理的部分）。

$$(\overline{\overline{A + B\overline{C}}})(\overline{D(\overline{E + \overline{F}})}) = (A + B\overline{C})(\overline{D(\overline{E + \overline{F}})})$$

步骤 4：为等式右边第二项应用狄摩根定理。

$$(A + B\overline{C})(\overline{D(\overline{E + \overline{F}})}) = (A + B\overline{C})(\overline{D} + (\overline{\overline{E + \overline{F}}}))$$

步骤 5：使用法则 9（$\overline{\overline{A}} = A$）消去 $E + \overline{F}$ 部分项上方的双重横杠。

$$(A + B\overline{C})(\overline{D} + \overline{\overline{E + \overline{F}}}) = (A + B\overline{C})(\overline{D} + E + \overline{F})$$

下面的三个例子将进一步解释怎样使用狄摩根定理。

例 4.5　对下面的表达式使用狄摩根定理。

(a) $\overline{(A + B + C)D}$

(b) $\overline{ABC + DEF}$

(c) $\overline{A\overline{B} + \overline{C}D + EF}$

解：(a) 让 $A + B + C = X$ 和 $D = Y$。表达式 $\overline{(A + B + C)D}$ 就具有了 $\overline{X + Y} = \overline{X}\overline{Y}$ 形式，因而可以写成

$$\overline{(A + B + C)D} = \overline{A + B + C} + \overline{D}$$

下一步，为前一项 $\overline{A + B + C}$ 应用狄摩根定理。

$$\overline{A + B + C} + \overline{D} = \overline{A}\,\overline{B}\,\overline{C} + \overline{D}$$

(b) 让 $ABC = X$，$DEF = Y$，表达式 $\overline{ABC + DEF}$ 就具有了 $\overline{X + Y} = \overline{X}\overline{Y}$ 形式，因而可以写成

$$\overline{ABC + DEF} = (\overline{ABC})(\overline{DEF})$$

下一步，为右边的项 $\overline{ABC}$ 和 $\overline{DEF}$ 应用狄摩根定理。

$$(\overline{ABC})(\overline{DEF}) = (\overline{A} + \overline{B} + \overline{C})(\overline{D} + \overline{E} + \overline{F})$$

(c) 让 $A\overline{B} = X$、$\overline{C}D = Y$ 和 $EF = Z$。表达式 $\overline{A\overline{B} + \overline{C}D + EF}$ 就具有了 $\overline{X + Y + Z} = \overline{X}\,\overline{Y}\,\overline{Z}$ 形式，因而可以写成

$$\overline{A\overline{B}+\overline{C}D+EF} = (\overline{A\overline{B}})(\overline{\overline{C}D})(\overline{EF})$$

下一步，为右边的项 $\overline{A\overline{B}}$、$\overline{\overline{C}D}$ 和 $\overline{EF}$ 应用狄摩根定理。

$$(\overline{A\overline{B}})(\overline{\overline{C}D})(\overline{EF}) = (\overline{A}+B)(C+\overline{D})(\overline{E}+\overline{F})$$

相关问题： 为表达式 $\overline{\overline{ABC}+D+E}$ 应用狄摩根定理。

例 4.6 对每个表达式使用狄摩根定理。

(a) $\overline{(\overline{A+B})+\overline{C}}$

(b) $\overline{(\overline{A}+B)+CD}$

(c) $\overline{(A+B)\overline{C}\overline{D}+E+\overline{F}}$

解： (a) $\overline{(\overline{A+B})+\overline{C}} = (\overline{\overline{A+B}})\,\overline{\overline{C}} = (A+B)C$

(b) $\overline{(\overline{A}+B)+CD} = (\overline{\overline{A}+B})\,\overline{CD} = (\overline{\overline{A}}\,\overline{B})(\overline{C}+\overline{D}) = A\overline{B}(\overline{C}+\overline{D})$

(c) $\overline{(A+B)\overline{C}\overline{D}+E+\overline{F}} = (\overline{(A+B)\overline{C}\overline{D}})(\overline{E+\overline{F}}) = (\overline{A}\,\overline{B}+C+D)\overline{E}F$

相关问题： 对表达式 $\overline{\overline{A}B(C+\overline{D})+E}$ 使用狄摩根定理。

例 4.7 异或门的布尔表达式为 $A\overline{B}+\overline{A}B$。用此作为开始点，使用狄摩根定理及任何其他可以使用的法则，推导出一个同或门的表达式。

解： 从异或门表达式的反相开始，然后应用狄摩根定理，如下所示：

$$\overline{A\overline{B}+\overline{A}B} = (\overline{A\overline{B}})(\overline{\overline{A}B}) = (\overline{A}+\overline{\overline{B}})(\overline{\overline{A}}+\overline{B}) = (\overline{A}+B)(A+\overline{B})$$

下一步，使用分配律和法则 8($A \cdot \overline{A}=0$)

$$(\overline{A}+B)(A+\overline{B}) = \overline{A}A+\overline{A}\,\overline{B}+AB+B\overline{B} = \overline{A}\,\overline{B}+AB$$

同或门的最终表达式为 $\overline{A}\,\overline{B}+AB$。注意，只要两个变量都为 0 或者都为 1，这个表达式就等于 1。

相关问题： 从 4 输入与非门的表达式开始，使用狄摩根定理推导出一个 4 输入非-或门的表达式。

4.3 节 温故而知新

1. 对如下的表达式应用狄摩根定理：

(a) $\overline{ABC}+(\overline{\overline{D}+E})$ (b) $\overline{(A+B)}\cdot\overline{C}$ (c) $\overline{A+B+C}+\overline{\overline{D}E}$

4.4 逻辑电路的布尔分析

4.4.1 逻辑电路的布尔表达式

◇ 逻辑电路可以用布尔等式来描述。

为了推导出一个给定逻辑电路的布尔表达式，从最左面的输入开始一直到最后的输出，写出每个门的表达式。对于图 4.16 中的示例电路，布尔表达式的确定如下所示：

1. 最左面输入 C 和 D 的与门的表达式为 CD。
2. 最左面与门的输出是或门的一个输入而 B 是另一个输入，所以或门的表达式为 $B+CD$。
3. 或门的输出是最右边与门的一个输入而 A 是另一个输入，所以该与门的表达式为 $A(B+CD)$，这就是整个电路最终的表达式。

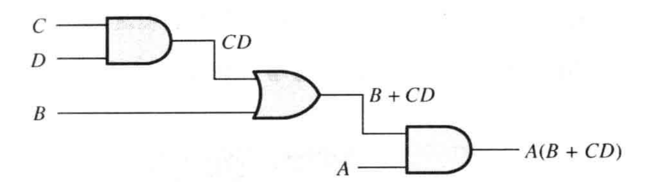


图 4.16　一个逻辑电路，给出了输出布尔表达式的推导过程

4.4.2　构建逻辑电路的真值表

◇ 逻辑电路可以用真值表来描述。

一旦给定的逻辑电路的布尔表达式得到确定，就可以构建其真值表，真值表用来展示输入变量的所有可能值所对应的输出。这个过程需要为输入变量所有可能数值的组合来评估布尔表达式。对于图 4.16 所示的电路，有 4 个输入变量（A、B、C 和 D），所以有 16（$2^4=16$）种可能的数值组合。

表达式的估算　为了对表达式 $A(B+CD)$ 做出评估，首先使用布尔加法和乘法法则，寻找使得表达式等于 1 的变量的值。在这种情况下，只有当 $A=1$ 及 $B+CD=1$ 的时候，表达式才等于 1，因为

$$A(B+CD)=1\cdot 1=1$$

现在确定什么时候 $B+CD$ 项等于 1。如果 $B=1$ 或者 $CD=1$，或者 B 和 CD 都等于 1，那么 $B+CD$ 项就等于 1，因为

$$B+CD=1+0=1$$
$$B+CD=0+1=1$$
$$B+CD=1+1=1$$

只有 $C=1$ 和 $D=1$ 时，CD 项才等于 1。

总结，当 $A=1$ 及 $B=1$ 而不用管 C 和 D 的数值，或者 $A=1$ 及 $C=1$ 并且 $D=1$ 而不用管 B 的数值时，表达式 $A(B+CD)=1$。对于变量的所有其他数值组合，表达式 $A(B+CD)=0$。

结果放入真值表　第一步，如表 4.5 所示，按照二进制的序列把输入变量 0 和 1 的 16 种组合列出来。下一步，对于在估算中已经确定的输入变量的每一种组合，在对应的输出列上写上 1。对于所有其他的输入变量组合，在相应的输出列中写上 0。这些结果如表 4.5 所示。

表 4.5　图 4.16 中逻辑电路的真值表

输入				输出
A	B	C	D	$A(B+CD)$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

例 4.8　使用 Multisim 仿真软件为图 4.16 的逻辑电路产生一个真值表。

解：在 Multisim 里画出电路，然后将 Multisim 逻辑转换器连接到输入和输出，如图 4.17 所示。点击 [→ 1011] 转换菜单框，如显示的那样会出现一个真值表。

点击 10|1 SIMP A|B 菜单框，还可以产生一个简单的布尔表达式。

相关问题：打开 Multisim 仿真软件创建电路，如例题那样进行转换。

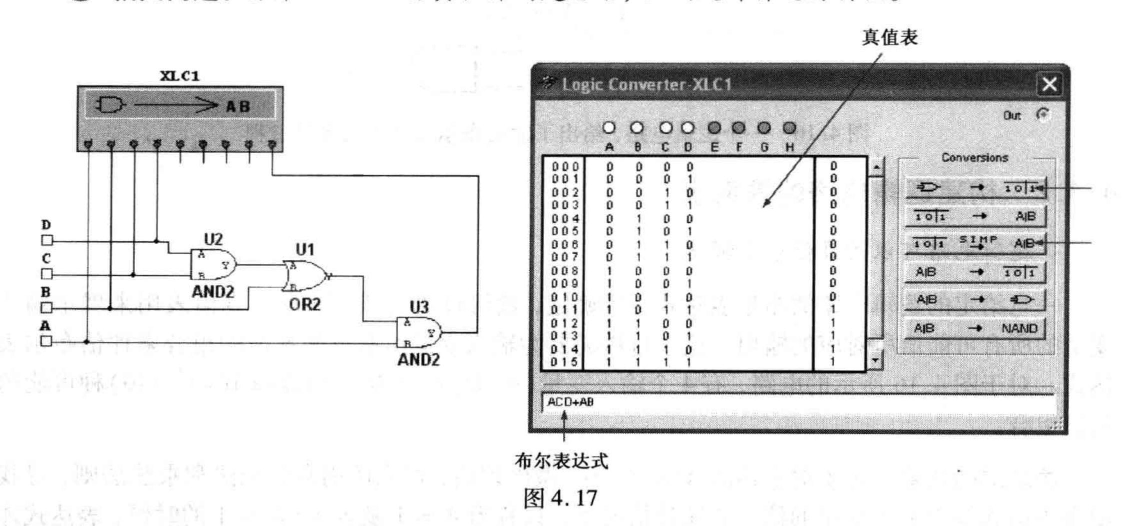


图 4.17

4.4 节 温故而知新

1. 在图 4.16 中，使用或门替代与门，使用与门替代或门，然后写出输出的布尔表达式。
2. 为问题 1 的电路创建真值表。

4.5 使用布尔代数进行化简

简化的布尔表达式使用尽可能少的门，从而实现给定表达式的功能。例 4.9 到例 4.12 给出了布尔代数的化简。

例 4.9 使用布尔代数，化简下面的表达式：$AB + A(B + C) + B(B + C)$。

解：下面所给出的步骤不一定是唯一的方法。

步骤 1：为表达式中的第二项和第三项应用分配律，如下所示：

$$AB + AB + AC + BB + BC$$

步骤 2：为等式右边第四项应用法则 7（$BB = B$）。

$$AB + AB + AC + B + BC$$

步骤 3：为等式右边第三项应用法则 5（$AB + AB = AB$）。

$$AB + AC + B + BC$$

步骤 4：为等式右边最后两项应用法则 10（$B + BC = B$）。

$$AB + AC + B$$

步骤 5：为等式右边第一项和第三项应用法则 10（$AB + B = B$）。

$$B + AC$$

在这里，表达式已被尽可能地简化了。一旦在布尔代数的应用中获得了经验，就可以经常合并许多独立的步骤。

相关问题：化简布尔表达式 $A\overline{B} + A(\overline{B + C}) + B(\overline{B + C})$。

图 4.18 展示了示例 4.9 中的化简过程，它大幅度地减少了实现该表达式的逻辑门的个数。

图 4.7(a)展示了在原来形式下需要 5 个门来实现该表达式；然而简化的表达式只需要两个门，如图 4.7(b)所示。意识到这两个门电路是等价的，这一点是非常重要的。也就是说，A、B、C 输入的任意电平组合，从这两个电路得到的都是一样的输出。化简意味着用更少的门实现一样的功能。

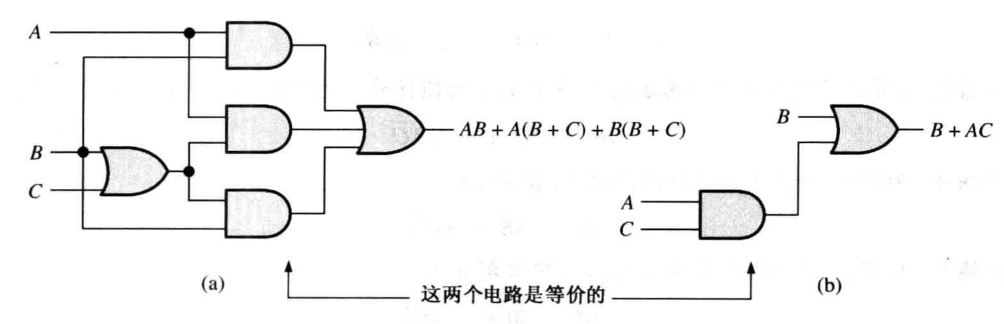


图 4.18　例 4.9 的门电路。打开文件 F04-18 检验等价性

例 4.10　化简下面的布尔表达式：

$$[A\overline{B}(C + BD) + \overline{A}\,\overline{B}]C$$

注意：中括号和圆括号表示一样的意思：里面的项乘以(与)外面的项。

解：步骤 1：为中括号里面的项应用分配律。

$$(A\overline{B}C + A\overline{B}BD + \overline{A}\,\overline{B})C$$

步骤 2：为等式右边圆括号里面的第二项应用法则 8($\overline{B}B = 0$)。

$$(A\overline{B}C + A \cdot 0 \cdot D + \overline{A}\,\overline{B})C$$

步骤 3：为等式右边圆括号里面的第二项应用法则 3($A \cdot 0 \cdot D = 0$)。

$$(A\overline{B}C + 0 + \overline{A}\,\overline{B})C$$

步骤 4：为等式右边圆括号内部应用法则 1(舍去 0)。

$$(A\overline{B}C + \overline{A}\,\overline{B})C$$

步骤 5：为等式右边圆括号内部应分配律。

$$A\overline{B}CC + \overline{A}\,\overline{B}C$$

步骤 6：对等式右边第一项应用法则 7($CC = C$)。

$$A\overline{B}C + \overline{A}\,\overline{B}C$$

步骤 7：对等式右边进行因式分解，分解出 $\overline{B}C$ 。

$$\overline{B}C(A + \overline{A})$$

步骤 8：对等式右边应用法则 6($A + \overline{A} = 1$)。

$$\overline{B}C \cdot 1$$

步骤 9：对等式右边应用法则 4(舍去 1)。

$$\overline{B}C$$

相关问题：化简布尔表达式 $[AB(C + \overline{BD}) + \overline{AB}]CD$。

例 4.11　化简下面的布尔表达式：

$$\overline{A}BC + A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}C + ABC$$

解：步骤1： 从第一项和最后一项中分解出 BC。

$$BC(\overline{A} + A) + A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}C$$

步骤2： 对等式右边圆括号中的项应用法则 6($\overline{A} + A = 1$)，并且从第二项和最后一项中分解出 $A\overline{B}$。

$$BC \cdot 1 + A\overline{B}(\overline{C} + C) + \overline{A}\,\overline{B}\,\overline{C}$$

步骤3： 为等式右边第一项应用法则4(舍去1)，为圆括号中的项应用法则6($\overline{C} + C = 1$)。

$$BC + A\overline{B} \cdot 1 + \overline{A}\,\overline{B}\,\overline{C}$$

步骤4： 为等式右边第二项应用法则4(舍去1)。

$$BC + A\overline{B} + \overline{A}\,\overline{B}\,\overline{C}$$

步骤5： 从等式右边的第二项和第三项中分解出 $\overline{B}$。

$$BC + \overline{B}(A + \overline{A}\,\overline{C})$$

步骤6： 为等式右边圆括号中的项应用法则 11($A + \overline{A}\,\overline{C} = A + \overline{C}$)。

$$BC + \overline{B}(A + \overline{C})$$

步骤7： 对等式右边使用分配律和交换律得到下面的表达式：

$$BC + A\overline{B} + \overline{B}\,\overline{C}$$

相关问题： 化简布尔表达式 $AB\overline{C} + \overline{A}\overline{B}C + \overline{A}BC + \overline{A}\overline{B}\overline{C}$ 。

例4.12 化简下面的布尔表达式：

$$\overline{AB + AC} + \overline{A}\,\overline{B}C$$

解：步骤1： 为第一项应用狄摩根定理。

$$(\overline{AB})(\overline{AC}) + \overline{A}\,\overline{B}C$$

步骤2： 为等式右边圆括号中的每一项都应用狄摩根定理。

$$(\overline{A} + \overline{B})(\overline{A} + \overline{C}) + \overline{A}\,\overline{B}C$$

步骤3： 为等式右边圆括号中的两项应用分配律。

$$\overline{A}\,\overline{A} + \overline{A}\,\overline{C} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C} + \overline{A}\,\overline{B}C$$

步骤4： 为等式右边第一项应用法则 7($\overline{A}\,\overline{A} = \overline{A}$)，然后为第三项和最后一项应用法则 10 [$\overline{A}\,\overline{B} + \overline{A}\,\overline{B}C = \overline{A}\,\overline{B}(1 + C) = \overline{A}\,\overline{B}$]。

$$\overline{A} + \overline{A}\,\overline{C} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C}$$

步骤5： 为等式右边第一项和第二项应用法则 10[$\overline{A} + \overline{A}\,\overline{C} = \overline{A}(1 + \overline{C}) = \overline{A}$]。

$$\overline{A} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C}$$

步骤6： 为等式右边第一项和第二项应用法则 10[$\overline{A} + \overline{A}\,\overline{B} = \overline{A}(1 + \overline{B}) = \overline{A}$]。

$$\overline{A} + \overline{B}\,\overline{C}$$

相关问题： 化简布尔表达式 $\overline{AB} + \overline{AC} + \overline{A}\overline{B}\overline{C}$ 。

例4.13 使用 Multisim 仿真软件对图 4.18 的逻辑电路进行化简。

解：步骤1： 如图 4.19 所示，把电路与 Multisim 的逻辑转换器相连。

步骤2： 点击 → 10|1 菜单框，创建真值表。

步骤3： 点击 10|1 SIMP AIB 菜单框，化简布尔表达式。

步骤 4：点击 A|B → [门符号] 菜单框，化简逻辑电路。

相关问题：使用 Multisim 软件创建启动并完成如例题给出的逻辑化简。

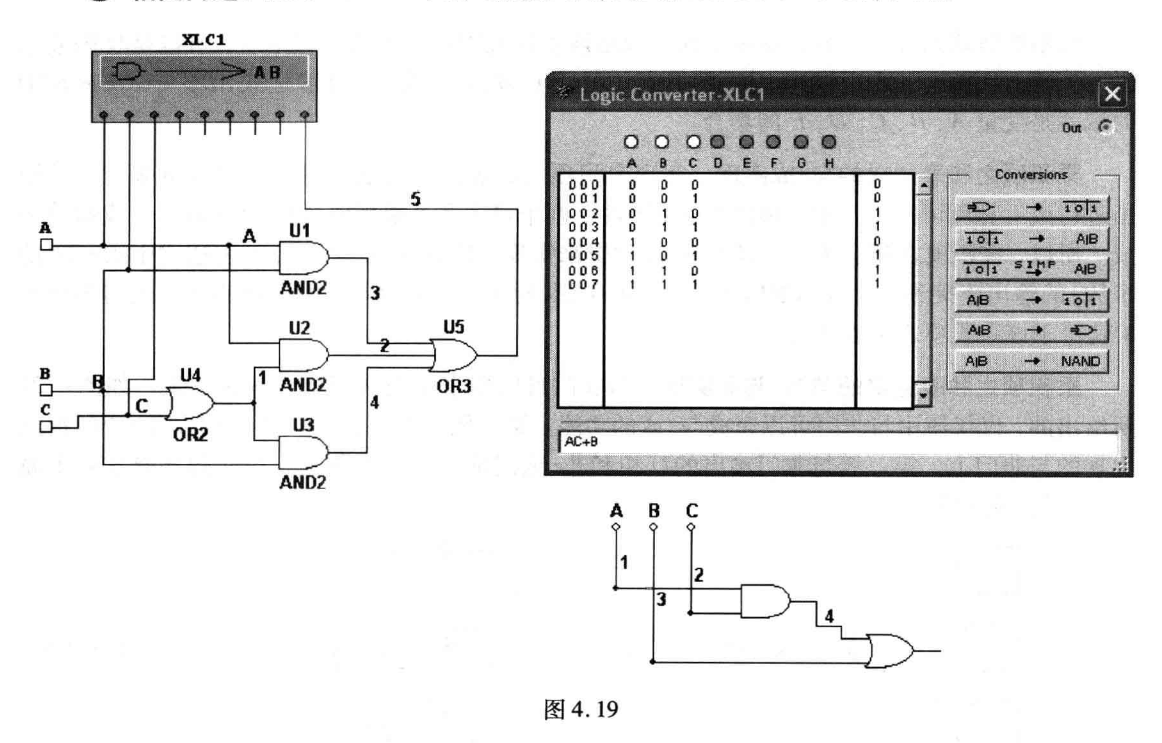


图 4.19

4.5 节　温故而知新

1. 尽可能地化简如下的布尔表达式：

   (a) $A + AB + A\bar{B}C$　　(b) $(\bar{A} + B)C + ABC$　　(c) $A\bar{B}C(BD + CDE) + A\bar{C}$

2. 使用和表达式中相对应的逻辑门来实现问题 1 中的表达式。然后化简表达式完成电路，比较所需的门的个数。

4.6　布尔表达式的标准形式

4.6.1　乘积项之和(SOP)形式

◇ 乘积项之和表达式可以由一个或门及两个或者更多的与门实现。

在 4.1 节中，乘积项定义为由文字(变量或者它们的反码)的乘积(布尔乘法)所组成的项。当两个或者更多个乘积项由布尔加法加起来时，得到的表达式就是乘积项之和(SOP，也称为与或表达式)。一些例子是

$$AB + ABC$$
$$ABC + CDE + \bar{B}C\bar{D}$$
$$\bar{A}B + \bar{A}B\bar{C} + AC$$

当然，一个乘积项之和表达式也可以包含单变量项，例如 $A + \bar{A}\,\bar{B}C + BC\bar{D}$。参见上一节的化简例子，就会发现每一个最终表达式都是单个的乘积项或者乘积项之和的形式。在乘积项之和表

达式中，单个上划线不能延伸到一个以上的变量；但是，一个项中的多个变量可以分别拥有一个上划线。例如，可以有 $\bar{A}\bar{B}\bar{C}$ 项，而不是 $\overline{ABC}$ 。

布尔表达式的域 一般布尔表达式的域是该表达式中所包含的变量集合，可以是反码或原码的形式。例如，表达式 $\bar{A}B + A\bar{B}C$ 的域就是变量 A、B、C 的集合，而表达式 $AB\bar{C} + C\bar{D}E + \bar{B}C\bar{D}$ 的域就是变量 A、B、C、D、E 的集合。

乘积项之和表达式的与/或实现 实现乘积项之和表达式仅仅需将两个或者更多与门的输出进行或运算就可以了。乘积项由与运算产生，两个或者多个乘积项的和(加)由一个或运算完成。所以，乘积项之和表达式可以由与-或逻辑来实现，其中与门(个数等于表达式中乘积项的个数)的输出连接到一个或门的输入上，如图 4.20 所示的 $AB + BCD + AC$ 表达式。或门的输出 X 就等于该乘积项之和表达式。

乘积项之和表达式的与非/与非实现 与非门可以用来实现乘积项之和表达式。如图 4.21 所给出的，仅仅使用与非门可以完成与/或的功能。第一级与非门的输出连接到以非-或门形式出现的与非门上。第一级与非门输出的反相和非-或门输入的反相抵消，最后的结果实际上就是一个与/或电路。

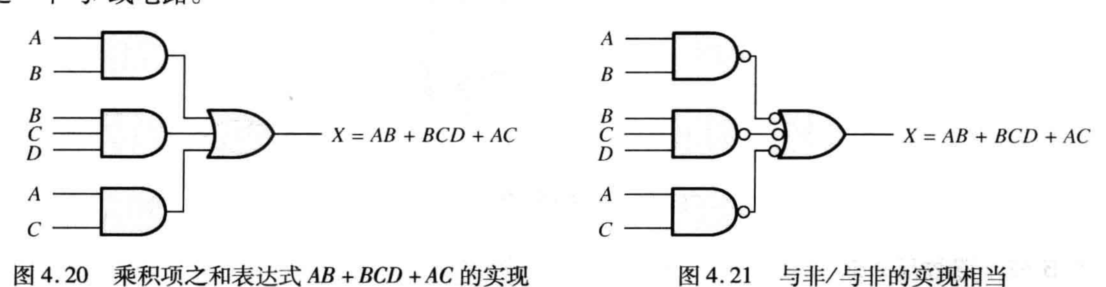


图 4.20 乘积项之和表达式 $AB + BCD + AC$ 的实现

图 4.21 与非/与非的实现相当于图4.20中的与/或

4.6.2 一般表达式向乘积项之和形式的转换

应用布尔代数技术，任何的逻辑表达式都可以转换为乘积项之和形式。例如，表达式 $A(B + CD)$ 就可以使用分配律而转换为乘积项之和形式：

$$A(B + CD) = AB + ACD$$

例 4.14 把下面的每个布尔表达式转换为乘积项之和形式：

(a) $AB + B(CD + EF)$ (b) $(A + B)(B + C + D)$ (c) $\overline{\overline{(A + B)} + C}$

解：(a) $AB + B(CD + EF) = AB + BCD + BEF$

(b) $(A + B)(B + C + D) = AB + AC + AD + BB + BC + BD$

(c) $\overline{\overline{(A + B)} + C} = \overline{\overline{(A + B)}}\bar{C} = (A + B)\bar{C} = A\bar{C} + B\bar{C}$

相关问题：把 $\bar{A}B\bar{C} + (A + \bar{B})(B + \bar{C} + A\bar{B})$ 转换为乘积项之和的形式。

4.6.3 最小项(标准乘积项)之和形式

到目前为止，已经介绍了乘积项之和表达式，其中有一些乘积项并没有包含表达式域中的所有变量。例如，表达式 $\bar{A}B\bar{C} + A\bar{B}D + \bar{A}\bar{B}\bar{C}D$ 的域由变量 A、B、C 和 D 组成。但是，注意在表达式的前两项中并没有出现域的全部变量的集合：也就是说，第一项缺少了 D 或 $\bar{D}$，第二项缺少了 C 或 $\bar{C}$ 。

最小项(标准乘积项)之和表达式的每一个乘积项都包含该表达式域中的所有变量。例如，$A\overline{B}CD+\overline{A}\,\overline{B}C\overline{D}+AB\overline{C}\,\overline{D}$就是一个最小项之和表达式。最小项之和表达式在构建真值表(在4.7节介绍)时是非常重要的，并且在卡诺图的化简(在4.8节介绍)中也是非常重要的。任何非标准的乘积项之和表达式(简单看成乘积项之和)都可以使用布尔代数把它变换为最小项之和形式。

把乘积项转换为最小项　乘积项之和表达式中没有包含域的所有变量的每一个乘积项，都可以扩展成最小项以包含域中所有的变量或反变量。如下面的步骤所表述的一样，使用表4.1中的布尔代数法则6($A+\overline{A}=1$)：变量加上它的反码等于1，可以把非标准乘积项表达式转换为最小项表达式。

步骤1： 对每个非标准乘积项乘上一项，这一项由此乘积项中没有出现的变量加上它的反变量组成。这样就产生了两个乘积项。如所知道的那样，任何乘积项乘以1，它的值不变。

步骤2： 重复步骤1直到所有的乘积项都成为最小项(乘积项中包含了域中所有的原变量或它的反变量，也称为标准乘积项)。在把乘积项变为最小项的过程中，每一个乘积项补上一个缺少的变量就变成了两项，如例4.15所给出的。

例4.15　把下面的布尔表达式转换成最小项的形式：

$$A\overline{B}C+\overline{A}\,\overline{B}+AB\overline{C}D$$

解： 乘积项表达式的域是A、B、C、D。每次取一项，第一项$A\overline{B}C$缺少变量D或$\overline{D}$，所以用$D+\overline{D}$乘第一项，如下所示：

$$A\overline{B}C=A\overline{B}C(D+\overline{D})=A\overline{B}CD+A\overline{B}C\overline{D}$$

在此情况下，结果就是两个最小项。

第二项$\overline{A}\,\overline{B}$缺少变量$C$或$\overline{C}$及$D$或$\overline{D}$，所以先用$C+\overline{C}$乘第二项，如下所示：

$$\overline{A}\,\overline{B}=\overline{A}\,\overline{B}(C+\overline{C})=\overline{A}\,\overline{B}C+\overline{A}\,\overline{B}\,\overline{C}$$

上式得到的两个乘积项缺少D或$\overline{D}$，所以用$D+\overline{D}$乘这两项，如下所示：

$$\begin{aligned}\overline{A}\,\overline{B}&=\overline{A}\,\overline{B}C+\overline{A}\,\overline{B}\,\overline{C}=\overline{A}\,\overline{B}C(D+\overline{D})+\overline{A}\,\overline{B}\,\overline{C}(D+\overline{D})\\&=\overline{A}\,\overline{B}CD+\overline{A}\,\overline{B}C\overline{D}+\overline{A}\,\overline{B}\,\overline{C}D+\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\end{aligned}$$

在此例子中，结果得到四个最小项。

第三项$AB\overline{C}D$已经是最小项了。原始表达式的最小项之和的完整表达式如下：

$$A\overline{B}C+\overline{A}\,\overline{B}+AB\overline{C}D=A\overline{B}CD+A\overline{B}C\overline{D}+\overline{A}\,\overline{B}CD+\overline{A}\,\overline{B}C\overline{D}+\overline{A}\,\overline{B}\,\overline{C}D+\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}+AB\overline{C}D$$

相关问题： 把表达式$W\overline{X}Y+\overline{X}Y\overline{Z}+WX\overline{Y}$转换为最小项之和的形式。

最小项的二进制表示　一个最小项等于1，这时仅对应一种二进制变量值的组合。例如，乘积项$A\overline{B}C\overline{D}$等于1，这时$A=1$，$B=0$，$C=1$，$D=0$，而对于其他二进制变量值的组合都等于0。

$$A\overline{B}C\overline{D}=1\cdot\overline{0}\cdot1\cdot\overline{0}=1\cdot1\cdot1\cdot1=1$$

其中，乘积项的二进制数是1010(十进制数10)。

记住，乘积项是由与门实现的，只有在与门的输入都是1的情况下输出才为1。反相器用来产生所需要的变量的反码。

一个乘积项之和的表达式等于1，只有在表达式中有一个或多个乘积项等于1时才成立。

例4.16 已知下面的最小项之和表达式等于1，确定每个最小项的二进制值：

$$ABCD + A\overline{B}\overline{C}D + \overline{A}\overline{B}\overline{C}\overline{D}$$

解：$ABCD$ 项等于1，这时 $A = 1, B = 1, C = 1, D = 1$。

$$ABCD = 1 \cdot 1 \cdot 1 \cdot 1 = 1$$

$A\overline{B}\overline{C}D$ 项等于1，这时 $A = 1, B = 0, C = 0, D = 1$。

$$A\overline{B}\overline{C}D = 1 \cdot \overline{0} \cdot \overline{0} \cdot 1 = 1 \cdot 1 \cdot 1 \cdot 1 = 1$$

$\overline{A}\overline{B}\overline{C}\overline{D}$ 项等于1，这时 $A = 0, B = 0, C = 0, D = 0$。

$$\overline{A}\overline{B}\overline{C}\overline{D} = \overline{0} \cdot \overline{0} \cdot \overline{0} \cdot \overline{0} = 1 \cdot 1 \cdot 1 \cdot 1 = 1$$

乘积项之和的表达式等于1，只有在表达式中有一个或多个乘积项等于1时才成立。

相关问题：已知下面的最小项之和表达式等于1，确定每个最小项的二进制值：

$$\overline{X}YZ + X\overline{Y}Z + XY\overline{Z} + \overline{X}Y\overline{Z} + XYZ$$

这是否是最小项之和表达式？

4.6.4 和项之乘积(POS)形式

在4.1节中，和项定义为由文字(变量或者它们的反码)的和(布尔加)项。当两个或者更多的和项相乘时，结果表达式就是和项之乘积(POS，也称为或与表达式)。例如：

$$(\overline{A} + B)(A + \overline{B} + C)$$
$$(\overline{A} + \overline{B} + \overline{C})(C + \overline{D} + E)(\overline{B} + C + D)$$
$$(A + B)(A + \overline{B} + C)(\overline{A} + C)$$

一个和项之乘积表达式也可以包含单变量项，例如 $\overline{A}(A + \overline{B} + C)(\overline{B} + \overline{C} + D)$ 。在和项之乘积表达式中，一个单一的上划线不能扩展到多于一个的变量。例如，一个和项之乘积表达式中可以有 $\overline{A} + \overline{B} + \overline{C}$ 项，但是不可以有 $\overline{A + B + C}$ 项。

和项之乘积的实现 实现和项之乘积表达式，只需简单地把两个或多个或门的输出进行与运算。和项由或运算产生，两个或多个和项的乘积由一个与运算产生。因此，一个和项之乘积表达式可以由若干个或门(与表达式中和项的数目相同)的输出连接到一个与门的输入这样的逻辑电路来实现。图4.22给出了实现表达式 $(A + B)(B + C + D)(A + C)$ 的电路。与门的输出 X 等于和项之乘积表达式。

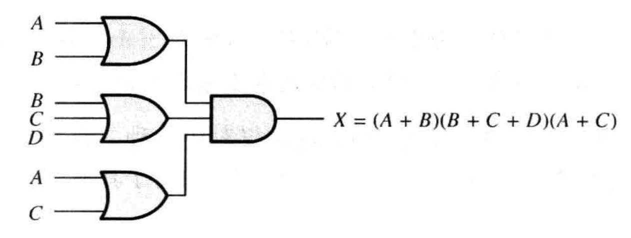


图4.22 和项之乘积表达式 $(A + B)(B + C + D)(A + C)$ 的实现

4.6.5 最大项(标准和项)之乘积形式

到目前为止，已经看到了和项之乘积表达式，其中有些和项并没有包含表达式域中所有变

量，例如，表达式

$$(A+\overline{B}+C)(A+B+\overline{D})(A+\overline{B}+\overline{C}+D)$$

它的域由变量 A、B、C、D 组成。注意，在表达式的前两项没有出现域中全部变量的集合，也就是说，第一项缺少了 D 或者 $\overline{D}$，第二项缺少了 C 或者 $\overline{C}$。

最大项之乘积表达式中的每一个和项都包含域中的所有变量。例如，表达式

$$(\overline{A}+\overline{B}+\overline{C}+\overline{D})(A+\overline{B}+C+D)(A+B+\overline{C}+D)$$

就是一个最大项之乘积表达式。使用布尔代数，任何非最大项之乘积表达式（简单地看成和项之乘积）可以转换成最大项之乘积表达式。

和项转换为最大项（标准和项） 在和之乘积表达式中没有包含域中所有变量的每一个和项，都可以扩展到最大项之乘积的形式，以包括域中的所有变量及它们的反码。正如在下面的步骤中所陈述的，可以使用表4.1中的布尔代数法则8（$A\cdot\overline{A}=0$）：一个变量乘以它的反码等于0，把和项之乘积表达式转换为最大项之积形式。

步骤1： 为每一个非标准和项加上一个由缺少的变量和其反码的乘积构成的项。结果就产生两个和项。如所知道的那样，在任意数上加0都不会改变这个数的值。

步骤2： 应用表4.1中的法则12：$[A+BC=(A+B)(A+C)]$。

步骤3： 重复步骤1直到所有的和项都包含域中的所有变量，这些变量可以是原码或者反码。

例4.17 把下面的布尔表达式转换为最大项之乘积形式：

$$(A+\overline{B}+C)(\overline{B}+C+\overline{D})(A+\overline{B}+\overline{C}+D)$$

解： 表达式的域是 A、B、C、D。每次取一个项，第一项 $A+\overline{B}+C$，缺少了 D 或者 $\overline{D}$，所以添加 $D\overline{D}$ 并使用法则12，如下所示：

$$A+\overline{B}+C=A+\overline{B}+C+D\overline{D}=(A+\overline{B}+C+D)(A+\overline{B}+C+\overline{D})$$

第二项缺少了变量 A 或者 $\overline{A}$，所以加上 $A\overline{A}$ 并且应用法则12，如下所示：

$$\overline{B}+C+\overline{D}=\overline{B}+C+\overline{D}+A\overline{A}=(A+\overline{B}+C+\overline{D})(\overline{A}+\overline{B}+C+\overline{D})$$

第三项 $A+\overline{B}+\overline{C}+D$ 已经是最大项了。原始表达式的最大项之乘积表达式如下所示：

$$(A+\overline{B}+C)(\overline{B}+C+\overline{D})(A+\overline{B}+\overline{C}+D)=$$
$$(A+\overline{B}+C+D)(A+\overline{B}+C+\overline{D})(A+\overline{B}+C+\overline{D})(\overline{A}+\overline{B}+C+\overline{D})(A+\overline{B}+\overline{C}+D)$$

相关问题： 把表达式 $(A+\overline{B})(B+C)$ 转换为最大项之乘积形式。

最大项的二进制表示 只有一种变量值的组合可以满足最大项等于0。例如，当 $A=0$、$B=1$、$C=0$ 及 $D=1$ 时，最大项 $A+\overline{B}+C+\overline{D}$ 等于0，而所有其他变量的值组合都是1。

$$A+\overline{B}+C+\overline{D}=0+\overline{1}+0+\overline{1}=0+0+0+0=0$$

其中，和项的二进制数是0101（十进制数5）。记住，和项由一个或门实现，只有或门的每一个输入都是0时，输出才是0。反相器用来产生所需变量的反码。

只有当表达式中的一个或者多个和项等于0时，和之乘积表达式才等于0。

例4.18 确定变量的二进制数，使得下面的最大项之和表达式等于0：

$$(A+B+C+D)(A+\overline{B}+\overline{C}+D)(\overline{A}+\overline{B}+\overline{C}+\overline{D})$$

解： 当 $A=0$、$B=0$、$C=0$ 和 $D=0$ 时，项 $A+B+C+D$ 等于0。

$$A+B+C+D=0+0+0+0=0$$

当 $A=0$、$B=1$、$C=1$ 和 $D=0$ 时，项 $A+\overline{B}+\overline{C}+D$ 等于0。

$$A+\overline{B}+\overline{C}+D=0+\overline{1}+\overline{1}+0=0+0+0+0=0$$

当 $A=1$、$B=1$、$C=1$ 和 $D=1$ 时，项 $\overline{A}+\overline{B}+\overline{C}+\overline{D}$ 等于0。

$$\overline{A}+\overline{B}+\overline{C}+\overline{D}=\overline{1}+\overline{1}+\overline{1}+\overline{1}=0+0+0+0=0$$

当3个最大项中的任何一项等于0时，该最大项表达式就等于0。

相关问题： 确定变量的二进制数，使得下面的和项之乘积表达式等于0：

$$(X+\overline{Y}+Z)(\overline{X}+Y+Z)(X+Y+\overline{Z})(\overline{X}+\overline{Y}+\overline{Z})(X+\overline{Y}+\overline{Z})$$

这是一个最大项表达式吗？

4.6.6 把最小项之和转换为最大项之积

给定的最小项之和表达式中乘积项的二进制数，并不会出现在对应的最大项之乘积表达式中。同样，最小项之和表达式中没有出现的二进制数却出现在了对应的最大项之乘积表达式中。因此，为了把最小项之和转换为最大项之乘积，可以采取下面的步骤：

步骤1： 计算乘积项之和表达式中的每一个乘积项。也就是说，确定表示乘积项的二进制数。

步骤2： 确定步骤1中的计算没有包含的所有二进制数。

步骤3： 为从步骤2得到的每一个二进制数写出相应的和项，并以和项之乘积形式表达。

使用类似的过程，就可以把和项之乘积转换为乘积项之和。

例4.19 把下面的最小项之和表达式转换为等价的最大项之乘积表达式：

$$\overline{A}\,\overline{B}\,\overline{C}+\overline{A}B\overline{C}+\overline{A}BC+A\overline{B}C+ABC$$

解： 计算如下所示：

$$000+010+011+101+111$$

因为表达式的域有3个变量，所有总共有 $8(2^3)$ 种可能的组合。最小项之和表达式包含这些组合中的5个，因此最大项之乘积表达式必须包含其余的3个，它们是001、100和110。记住，这些二进制数使得和项等于0。等价的乘积项之和表达式为

$$(A+B+\overline{C})(\overline{A}+B+C)(\overline{A}+\overline{B}+C)$$

相关问题： 通过在每个表达式中代入二进制数，可以验证这个例题中的最小项之和表达式与最大项之乘积表达式是等价的。

4.6节 温故而知新

1. 判断每个如下的表达式是乘积项之和表达式、最小项表达式、和项之乘积表达式还是最大项表达式。

   (a) $AB+\overline{A}BD+\overline{A}C\overline{D}$　(b) $(A+\overline{B}+C)(A+B+\overline{C})$

   (c) $\overline{A}BC+AB\overline{C}$　(d) $(A+\overline{C})(A+B)$

2. 把问题1的乘积项之和表达式转换为最小项表达式。
3. 把问题1的和项之乘积表达式转换为最大项表达式。

4.7　布尔表达式和真值表

4.7.1　把乘积项之和表达式转换为真值表的形式

在4.6节中，只有在至少有一个乘积项等于1时，乘积项之和表达式才等于1。真值表只是简单地列出了输入变量数值的可能组合及相应的输出数值(1或者0)。对于域有两个变量的表达式，那么这两个变量有4($2^2=4$)种不同的组合。对于域有3个变量的表达式，这些变量有8($2^3=8$)个不同的组合。对于域有4个变量的表达式，这些变量有16($2^4=16$)个不同的组合，以此类推。

构建真值表的第一步，是列出表达式中变量的所有可能的二进制数的组合。接下来，如果乘积项之和表达式不是最小项之和表达式，那么将其转换为最小项之和表达式。最后，对每个使得最小项之和表达式为1的二进制数，在其输出列(X)对应的位置放置一个1，为所有剩余的二进制数对应的输出位置放置0。这个过程如例4.20所示。

例4.20　为最小项之和表达式$\overline{A}\overline{B}C+A\overline{B}\overline{C}+ABC$写出一个真值表。

解：在这个域中有3个变量，所以有8个可能的变量的二进制数的组合，它们列在了表4.6左边的3列上。使得表达式中乘积项等于1的二进制数是：$\overline{A}\overline{B}C$:001，$A\overline{B}\overline{C}$:100，$ABC$:111。对于每一个这样的二进制数，在输出列的相应位置放置一个1，如表中所示。为剩余的二进制数的组合在输出列中放置0。

表4.6

输入			输出	
A	B	C	X	乘积项
0	0	0	0	
0	0	1	1	$\overline{A}\overline{B}C$
0	1	0	0	
0	1	1	0	
1	0	0	1	$A\overline{B}\overline{C}$
1	0	1	0	
1	1	0	0	
1	1	1	1	ABC

相关问题：为最小项之和表达式$\overline{A}B\overline{C}+A\overline{B}C$创建真值表。

4.7.2　把和项之乘积表达式转换为真值表的形式

回顾一下，只要有一个和项等于0，和项之乘积表达式就等于0。为了从和项之乘积表达式中构建真值表，可以像最小项之和表达式那样，列出所有可能的变量的二进制数组合，如果这个和项之乘积表达式不是最大项之乘积表达式，那么将该表达式转换为最大项之乘积表达式的形式。最后，使得最大项之乘积表达式等于0的二进制数对应的输出列(X)上放置一个0，为所有剩余的二进制数值对应的输出列上放置1。这个过程如例4.21所示。

例4.21　为下面的最大项之和表达式确定其真值表：

$$(A+B+C)(A+\overline{B}+C)(A+\overline{B}+\overline{C})(\overline{A}+B+\overline{C})(\overline{A}+\overline{B}+C)$$

解：这个域有3个变量，8个可能的二进制数列在表4.7左边的三列。使得表达式中和项的二进制数是：$A+B+C$:000；$A+\overline{B}+C$:010；$A+\overline{B}+\overline{C}$:011；$\overline{A}+B+\overline{C}$:101；$\overline{A}+\overline{B}+C$:110。为这些二进制数的每一个，在相应的输出列中都放置0，如表中所示。为其余的二进制数的组合，在相应的输出列中放置1。

表4.7

输入 A	B	C	输出 X	和项
0	0	0	0	$(A+B+C)$
0	0	1	1	
0	1	0	0	$(A+\overline{B}+C)$
0	1	1	0	$(A+\overline{B}+\overline{C})$
1	0	0	1	
1	0	1	0	$(\overline{A}+B+\overline{C})$
1	1	0	0	$(\overline{A}+\overline{B}+C)$
1	1	1	1	

注意这个例题中的真值表和例4.20中的真值表是一样的。这就表示前面例题中的乘积项之和表达式和此例中的和项之乘积表达式是等价的。

相关问题：为下面的最大项之乘积表达式开发真值表：

$$(A+\overline{B}+C)(A+B+\overline{C})(\overline{A}+\overline{B}+\overline{C})$$

4.7.3 从真值表确定标准表达式

为了确定真值表表示的最小项之和表达式，列出使得输出为1的输入变量的二进制数。把1替换成乘积项中相应的变量，0替换成反变量，这样就把每一个二进制数替换成相应的乘积项。例如，二进制数1010可以转换成如下所示的乘积项：

$$1010 \longrightarrow A\overline{B}C\overline{D}$$

代入以后，就会发现这个乘积项为1：

$$A\overline{B}C\overline{D} = 1\cdot\overline{0}\cdot 1\cdot\overline{0} = 1\cdot 1\cdot 1\cdot 1 = 1$$

为了确定真值表所表示的最大项之乘积表达式，列出使得输出为0的二进制数。把和项中的0替换成相应的变量，1替换成反变量，这样就把每一个二进制数替换成相应的和项。例如，二进制数1001可以转换成如下所示的和项：

$$1001 \longrightarrow \overline{A}+B+C+\overline{D}$$

代入以后，就会发现这个和项为0：

$$\overline{A}+B+C+\overline{D} = \overline{1}+0+0+\overline{1} = 0+0+0+0 = 0$$

例4.22 从表4.8中的真值表，确定最小项之和表达式及等价的最大项之乘积表达式。

解：在输出列中有4个1，并且相应的二进制数是011、100、110及111。把这些二进制数转换为乘积项，如下所示：

$$011 \longrightarrow \overline{A}BC$$
$$100 \longrightarrow A\overline{B}\,\overline{C}$$
$$110 \longrightarrow AB\overline{C}$$
$$111 \longrightarrow ABC$$

最后，输出 X 的最小项之和表达式是

$$X = \overline{A}BC + A\overline{B}\,\overline{C} + AB\overline{C} + ABC$$

把这些二进制数转换成和项，如下所示：

$$000 \longrightarrow A + B + C$$
$$001 \longrightarrow A + B + \overline{C}$$
$$010 \longrightarrow A + \overline{B} + C$$
$$101 \longrightarrow \overline{A} + B + \overline{C}$$

最后，输出 X 的最大项之乘积表达式是

$$X = (A + B + C)(A + B + \overline{C})(A + \overline{B} + C)(\overline{A} + B + \overline{C})$$

表 4.8

输入			输出
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

相关问题：通过代入二进制数，展示这个例题中所导出的最小项之和与最大项之乘积表达式是等价的，也就是对于任何一个二进制数，它们应当都是 1 或者都是 0，结果取决于该二进制数的值。

4.7 节　温故而知新

1. 如果一个布尔表达式有 5 个变量的变量域，那么它的真值表中有多少个二进制数？

2. 某个真值表中，对应二进制数 0110 的输出为 1。使用变量 W、X、Y 和 Z，把此二进制数转换为对应的和项。

3. 某个真值表中，对应二进制数 1100 的输出为 0。使用变量 W、X、Y 和 Z，把此二进制数转换为对应的和项。

4.8　卡诺图

◇ 卡诺图的目的是用于简化布尔表达式。

卡诺图（Karnaugh map）和真值表有相似之处，因为它呈现了所有可能的输入变量的数值，以及这些值所对应的输出。和真值表组织成列和行的样式不同，卡诺图是小方格的阵列，其中每一个小方格表示一个输入变量的二进制数。这种布置小方格的方式，使得对给定表达式的化简仅仅是把小方格恰当地进行组合的操作。卡诺图可以用于具有 2 个、3 个、4 个和 5 个变量的表达式，但是这里仅仅讨论 3 变量和 4 变量的情况，用以阐释该原理。4.10 节使用 32 个小方格的卡诺图来处理 5 个变量。

卡诺图中小方格的数目等于可能输入变量组合的总数目，也就是真值表中行的数目。对于 3 个变量，小方格的数目是 $2^3 = 8$。对于 4 个变量，小方格的数目为 $2^4 = 16$。

4.8.1　3 变量卡诺图

3 变量卡诺图是 8 个小方格阵列，如图 4.23(a) 所示。在这种情况下，A、B、C 用于表示变量。当然也可以使用其他字母。A 和 B 的二进制数沿着左边界（注意次序）而 C 的值横跨顶部。一个给定小方格的值就是同一行中左边的 A 和 B 的值及同一列中顶部 C 的值。例如，左上角小方格的二进制数为 000，

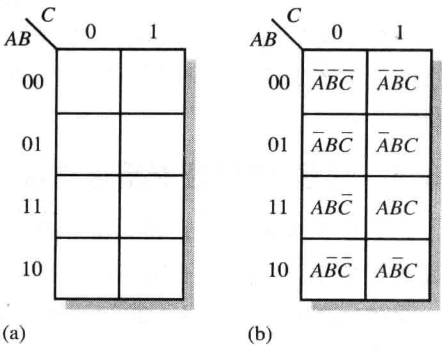


图 4.23　展示乘积项的 3 变量卡诺图

右下角小方格的二进制数为 101。图 4.21(b)给出了卡诺图中每一个小方格所表示的最小项。

4.8.2 4 变量卡诺图

4 变量卡诺图是 16 个小方格的阵列，如图 4.24(a)所示。A 和 B 的二进制数沿着左边界，C 和 D 的值横跨顶部。一个给定小方格的值就是同一行左边的 A 和 B 的值，以及同一列顶部的 C 和 D 的值。例如，右上角小方格的二进制数为 0010，右下角小方格的二进制数为 1010。图 4.24(b)展示了 4 变量卡诺图每一个单元所表示的最小项。

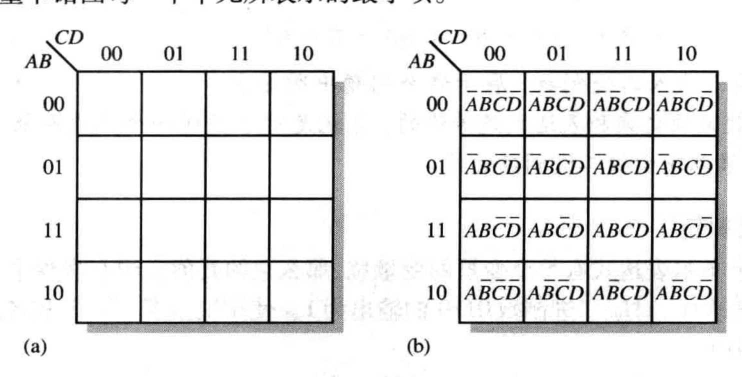


图 4.24 4 变量卡诺图

4.8.3 小方格相邻

◇ 仅有一个变量不同的小方格相邻。

◇ 有一个以上变量不同的小方格不相邻。

卡诺图中的小方格按一定方式布置，使得相邻的小方格之间只有一个变量发生变化。相邻(adjacency)定义为其中一个变量发生变化。在 3 变量图中 010 小方格和 000、011、110 小方格相邻，010 小方格和 001、111、100、101 小方格不相邻。

从物理布局上来看，每一个小方格都和与它的 4 条边紧挨着的小方格相邻。一个小方格和与它有对角接触的小方格不相邻。同样，在顶行的小方格和底部一行相对应的小方格相邻，左上角的小方格和右上角的小方格相邻。这称为"环绕"相邻，因为可以将这个图从顶部到底部环绕而形成一个圆柱体，或者从左到右环绕形成一个圆柱体。图 4.25 给出了 4 变量的卡诺图的小方格相邻，然而相邻法则同样适用于任意数目的小方格。

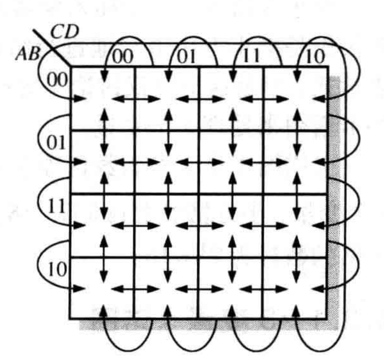


图 4.25 卡诺图上的相邻小方格仅有一个变量不同。在相邻的小方格之间由箭头标出

4.8.4 奎恩-麦克拉斯基化简法(Q-M 法)

对于 5 个及 5 个以上的变量，使用卡诺图最小化布尔表达式并不适用，实际上卡诺图仅适用于 4 个和 4 个以下的变量。同时，这种方法自身不能和计算机编程相匹配。

奎恩-麦克拉斯基(Quine-McClusky)化简法在化简多于 4 个或 5 个变量的逻辑函数时更为实用。它还具有一个优点，就是很容易用计算机或可编程计算器来实现它的工作。

奎恩-麦克拉斯基化简法在功能上和卡诺图相似，但是它的表格形式使其在使用计算机算法时更为有效，同时也给出了判断一个布尔函数是否已经最小化的方法。这个方法有时被看成为表格法。附录B介绍了奎恩-麦克拉斯基化简法。

4.8.5 Espresso 算法

尽管奎恩-麦克拉斯基化简法很适合用计算机编程来实现，可以比卡诺图处理更多的变量，但是在处理时间和存储容量的有效性上仍然不够理想。在函数中增加一个变量大致要增加一倍的参数，因此真值表的长度随着变量数目的增加呈指数形式递增。有大量变量的函数必须用其他的方法最小化，例如Espresso算法的逻辑最小化，事实上这已经成为一个标准。

和其他方法相比较，Espresso算法的几种量级次序方法在减少存储容量和计算时间方面很有效。基本上对变量和组合逻辑函数的与项数目没有限制。通常，具有几十个输出及几十个变量的函数都可以用Espresso算法来处理。

Espresso算法已经成为大多数可编程逻辑器件的逻辑合成工具中一个标准的逻辑函数化简步骤。对于一个多级的逻辑函数的实现，通过在目标装置(例如FPGA，即场可编程门阵列)中进行因式分解和变量的基本逻辑单元的映射最小化，使得结果得到优化。

4.8节 温故而知新

1. 在3变量卡诺图中，给出如下小方格表示的二进制数。
   (a)左上角 (b)右下角 (c)左下角 (d)右上角
2. 对于变量X、Y和Z，问题1中的每个小方格是如何对应于最小项的？
3. 对于4变量卡诺图重复问题1。
4. 对于4变量W、X、Y和Z的卡诺图，重复问题2。

4.9 卡诺图乘积项之和的最小化

4.9.1 最小项之和表达式的卡诺图映射

对于最小项之和表达式来说，为表达式中的每一个乘积项都在卡诺图上放置一个1。每一个1都放置在与乘积项数值相对应的小方格内。例如，对于乘积项$A\bar{B}C$，那么将一个1放置在3变量卡诺图的101小方格内。

当乘积项之和表达式被完全映射好之后，卡诺图上就会有一些1，其数目等于最小项之和表达式中乘积项的个数。不具有1的小方格就是表达式为0时所对应的小方格。通常，使用乘积项之和表达式时，0并不在图上绘出。下面的步骤及图4.26中的解释给出了映射卡诺图的过程。

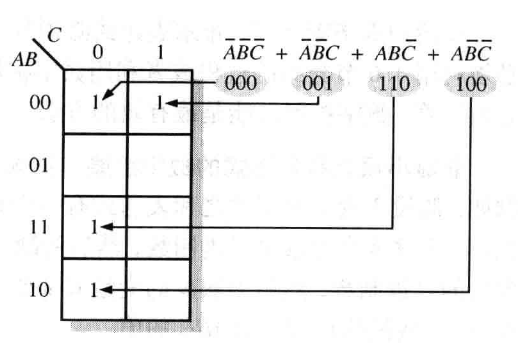


图4.26 最小项之和表达式映射的例子

步骤1： 确定最小项之和表达式中每一个乘积项的二进制数。通过练习，通常可以心算得到乘积项的值。

步骤2： 算完每一个乘积项之后，在卡诺图中和乘积项具有相同数值的小方格处放置1。

例 4.23 把下面的最小项之和表达式映射到卡诺图上。

$$\overline{A}\,\overline{B}C + \overline{A}B\overline{C} + AB\overline{C} + ABC$$

解： 表达式的计算如下所示。在图 4.27 的 3 变量卡诺图中，为表达式的每一个最小项都放置 1。

$$\begin{array}{cccc} \overline{A}\,\overline{B}C + & \overline{A}B\overline{C} + & AB\overline{C} + & ABC \\ 001 & 010 & 110 & 111 \end{array}$$

相关问题： 把最小项之和表达式 $\overline{A}BC + A\overline{B}C + A\overline{B}\overline{C}$ 映射到卡诺图上。

例 4.24 把下面的最小项之和表达式映射到卡诺图上。

$$\overline{A}\,\overline{B}CD + \overline{A}B\overline{C}\,\overline{D} + AB\overline{C}D + ABCD + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + A\overline{B}C\overline{D}$$

解： 表达式的计算如下所示。在图 4.28 的 4 变量卡诺图中，为表达式的每一个最小项都放置一个 1。

$$\begin{array}{ccccccc} \overline{A}\,\overline{B}CD + & \overline{A}B\overline{C}\,\overline{D} + & AB\overline{C}D + & ABCD + & AB\overline{C}\,\overline{D} + & \overline{A}\,\overline{B}\,\overline{C}D + & A\overline{B}C\overline{D} \\ 0011 & 0100 & 1101 & 1111 & 1100 & 0001 & 1010 \end{array}$$

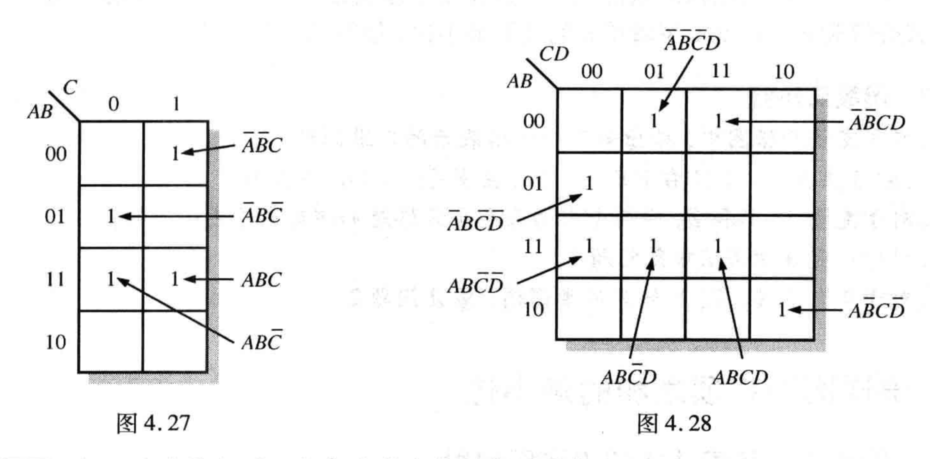


图 4.27　　　　图 4.28

相关问题： 把下面的最小项之和表达式映射到卡诺图上。

$$\overline{A}BC\overline{D} + ABC\overline{D} + AB\overline{C}\,\overline{D} + ABCD$$

4.9.2 非最小项(非标准乘积项)之和表达式的卡诺图映射

在使用卡诺图之前，布尔表达式必须处于标准形式。如果一个表达式为非标准形式，那么必须利用 4.6 节介绍的过程或者利用数字扩展，将其变换为标准形式。由于表达式必须在映射之前计算，数字扩展可能是最有效的方法。

非最小项之和表达式的数字扩展 回顾一下，非标准乘积项有一个或者多个缺少的变量。例如，假设 3 变量乘积项之和表达式有一个乘积项为 $A\overline{B}$，这个项可以数字扩展为标准形式。首先，写出这两个变量的二进制数，然后为缺少的变量 $\overline{C}$ 添加 0，得到 100。下一步，写出这两个变量的二进制数，然后为缺少的变量 C 添加 1，得到 101。最后得到的这两个二进制数就是最小项(标准乘积项) $A\overline{B}\overline{C}$ 和 $A\overline{B}C$ 的值。

作为另一个例子，假设 3 变量表达式的某个乘积项为 B(记住一个单变量可以在乘积项之和表达式中作为一个乘积项)。这一项可以数字扩展到标准项，如下所示。写出该变量的二进制数，然后添加缺少的变量 A 和 C 的所有可能值：

B
010
011
110
111

最后得到的 4 个二进制数就是最小项(标准乘积项) $\overline{A}B\overline{C}$、$\overline{A}BC$、$AB\overline{C}$、$ABC$ 的值。

例 4.25　把下面的乘积项之和表达式映射到卡诺图上。

$$\overline{A} + A\overline{B} + AB\overline{C}$$

解：这个表达式明显不是标准形式，因为并不是每一项都具有 3 个变量。第一项缺少两个变量，第二项缺少一个变量，第三项是标准的。首先将这些项进行数字扩展，如下所示：

$\overline{A}$	$+ A\overline{B}$	$+ AB\overline{C}$
000	100	110
001	101	
010		
011		

通过在图 4.29 的 3 变量卡诺图中相应的小方格内放置 1，对每一个二进制数进行映射。

相关问题：把乘积项之和表达式 $BC + \overline{A}\overline{C}$ 映射到卡诺图上。

例 4.26　把下面的乘积项之和表达式映射到卡诺图上。

$$\overline{B}\,\overline{C} + A\overline{B} + AB\overline{C} + A\overline{B}C\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + A\overline{B}CD$$

解：这个表达式明显不是标准形式，因为并不是每一项都具有 4 个变量。第一项和第二项都缺少两个变量，第三项缺少一个变量，其余项都是标准的。首先对这些项进行数字扩展，这包括缺少变量的所有组合，如下所示：

$\overline{B}\,\overline{C}$	$A\overline{B}$	$+ AB\overline{C}$	$+ A\overline{B}C\overline{D}$	$+ \overline{A}\,\overline{B}\,\overline{C}D$	$+ A\overline{B}CD$
0000	1000	1100	1010	0001	1011
0001	1001	1101			
1000	1010				
1001	1011				

通过在图 4.30 的 4 变量卡诺图中相应的小方格内放置 1，对每一个二进制数值进行映射。注意在扩展的表达式中有些值是重复的。

AB \ C	0	1
00	1	1
01	1	1
11	1	
10	1	1

图 4.29

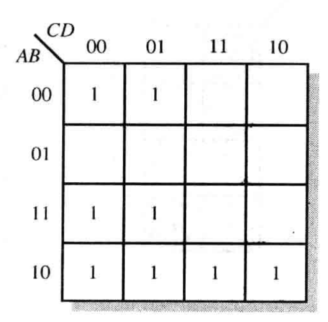


图 4.30

相关问题：把表达式 $A + \overline{C}D + AC\overline{D} + \overline{A}BC\overline{D}$ 映射到卡诺图上。

4.9.3 乘积项之和表达式的卡诺图化简

使得表达式包含尽可能少的项，每项包含尽可能少的变量，这个过程称为最小化。在乘积项之和表达式被映射之后，通过对1分组和从图上确定每组的最小乘积项来进行化简。(和项之乘积表达式的化简请参阅附录B。)

对1分组 根据下面的规则，将那些包含1的相邻小方格圈在一起，从而在卡诺图上对1进行分组。这样做的目的是使得每组(圈)尽可能大，而组尽可能少。

1. 一个小组只能含有1、2、4、8或者16个小方格，全都是2的幂。对于3变量卡诺图，最大的组是$2^3=8$个小方格。
2. 小组中的每一个小方格必须与相同小组内的一个或者多个小方格相邻，但是该小组中的所有小方格并不一定相互邻接。
3. 根据规则1，总是使得包含的1的数目尽可能多。
4. 图上的每一个1必须包含在至少一个组内。已经在某个组中的1，也可以包含在其他组中，只要两个重叠的组中都含有不属于其他组的1。

例4.27 把图4.31中每一个卡诺图上的1进行分组。

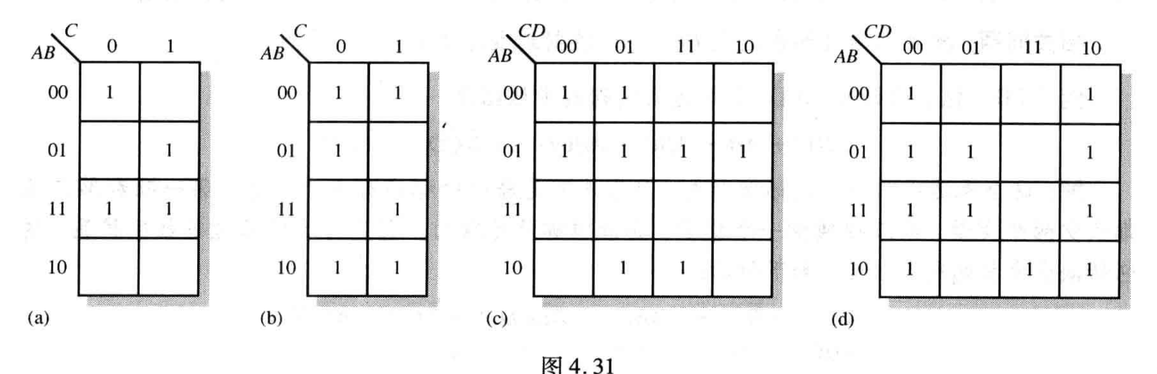


图4.31

解： 分组过程如图4.32所示。在某些情况下，可能具有不止一种方法对1进行分组以形成最大的组。

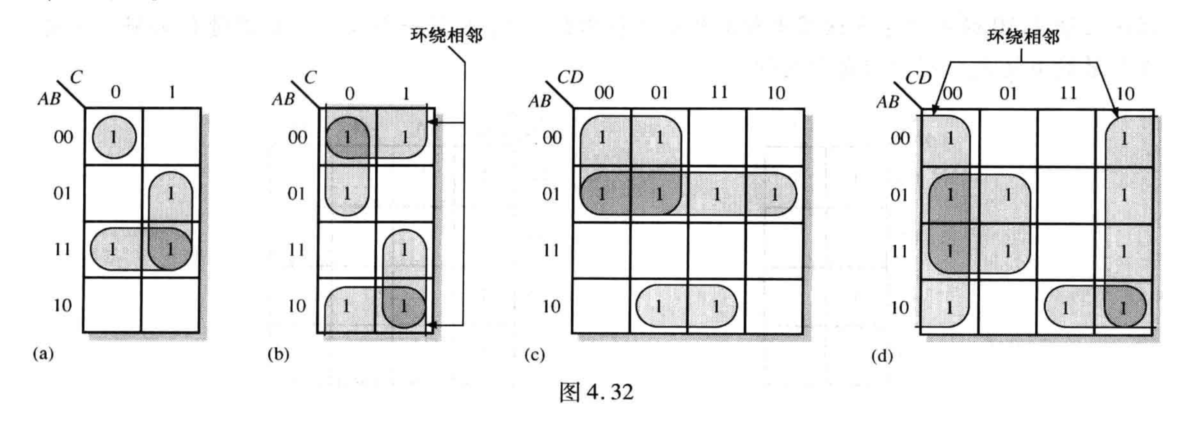


图4.32

相关问题： 对于图4.32所示的1，确定是否还有其他的方法进行分组，以得到最大分组情况下的最少的组。

从卡诺图上确定最小乘积项之和表达式　当表达式中所有表示最小项的1在卡诺图上得到恰当的映射和分组后，确定最小化的乘积项之和表达式的过程开始。应用下面的规则得到最小化的乘积项与最小化的和项的表达式。

1. 对有1的小方格进行分组。包含1的每一组小方格产生一个乘积项，组成这个乘积项的所有变量在这个组中仅出现一种形式(要么是原码，否则就是反码)。在这个组中如果变量发生了从原码到反码的变化，那么删除这些变量(这些变量称为矛盾变量)。
2. 为每个组确定最小乘积项。
   a. 3变量的卡诺图
      (1)一个小方格组产生3变量的乘积项
      (2)两个小方格组产生2变量的乘积项
      (3)四个小方格组产生1变量的项
      (4)八个小方格组产生表达式1
   b. 4变量的卡诺图
      (1)一个小方格组产生4变量的乘积项
      (2)两个小方格组产生3变量的乘积项
      (3)四个小方格组产生2变量的乘积项
      (4)八个小方格组产生1变量的项
      (5)十六个小方格组产生表达式1
3. 当从卡诺图推导出所有的最小乘积项后，把它们加起来形成最小乘积项之和表达式。

例4.28　为图4.33中的卡诺图确定乘积项，并写出最小乘积项之和表达式的结果。

解：把小组中原码和反码形式都有的变量删除。在图4.33中，8个小方格组的乘积项是B，因为在这组的小方格中包含了A和$\overline{A}$、C和$\overline{C}$及D和$\overline{D}$，这些已被删除。四个小方格组包含B和$\overline{B}$、D和$\overline{D}$，保留变量$\overline{A}$和C，形成乘积项$\overline{A}C$。两个小方格组包含B和$\overline{B}$，保留变量A、$\overline{C}$和D，形成乘积项$A\overline{C}D$。注意怎样通过重叠以使得组的规模最大。最后，最小乘积项之和表达式就是这些乘积项的和：

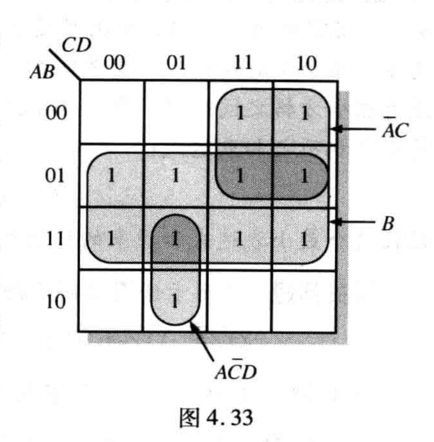


图4.33

$$B+\overline{A}C+A\overline{C}D$$

相关问题：对于图4.33中的卡诺图，在右下角小方格(1010)处添加一个1，然后确定乘积项之和表达式。

例4.29　为图4.34中的每一个卡诺图确定乘积项，并写出结果最小化的乘积项之和表达式。

解：每一个组得到的最小乘积项如图4.34所示。每个卡诺图的最小化的乘积项之和表达式如下所示：

(a) $AB+BC+\overline{A}\,\overline{B}\overline{C}$

(b) $\overline{B}+\overline{A}\overline{C}+AC$

(c) $\overline{A}B+\overline{A}\overline{C}+A\overline{B}D$

(d) $\overline{D}+A\overline{B}C+B\overline{C}$

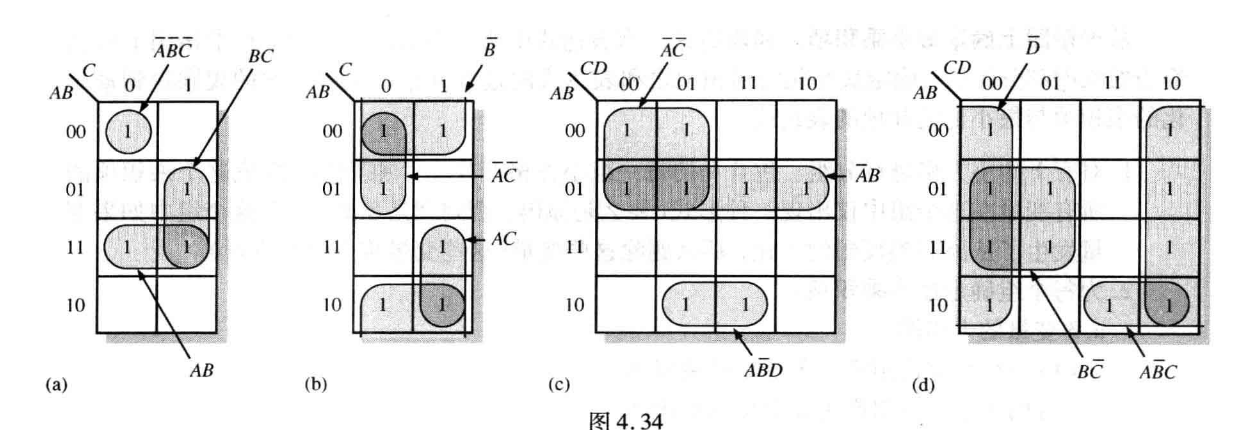


图 4.34

相关问题: 对图 4.34(d)的卡诺图,在小方格 0111 中放置 1,然后求出乘积项表达式。

例 4.30 使用卡诺图最小化下面的最小项之和表达式。

$$A\overline{B}C + \overline{A}BC + \overline{A}\,\overline{B}C + \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C}$$

解: 表达式的二进制数值是

$$101 + 011 + 001 + 000 + 100$$

最小项之和表达式的映射和分组后的小方格如图 4.35 所示。

注意那个“环绕”的四个小方格组,包括了顶部及底部中的所有 1。剩下的 1 归入两个小方格的重叠组中。4 个 1 的组产生一个单变量项 $\overline{B}$ 。通过观察该组内部,发现 B 是唯一一个没有在小方格之间改变的变量。两个 1 的组产生 2 变量项 $\overline{A}C$。通过观察该组内部,发现 $\overline{A}$ 和 C 没有在小方格之间改变。每一个小组的乘积项被展示出来,最后得到最小乘积项之和表达式:

$$\overline{B} + \overline{A}C$$

记住这个最小表达式和原来的标准表达式是等价的。

图 4.35

相关问题: 使用卡诺图化简下面的最小项表达式。

$$X\overline{Y}Z + XY\overline{Z} + \overline{X}YZ + \overline{X}\,\overline{Y}Z + X\overline{Y}\,\overline{Z} + XYZ$$

例 4.31 使用卡诺图化简下面的最小项表达式。

$$\overline{B}\,\overline{C}\,\overline{D} + \overline{A}B\overline{C}\,\overline{D} + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}CD + A\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + \overline{A}BC\overline{D} + ABC\overline{D} + A\overline{B}C\overline{D}$$

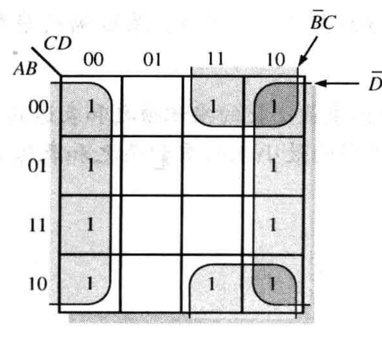


图 4.36

解: 第一项 $\overline{B}\,\overline{C}\,\overline{D}$ 必须扩展为 $A\overline{B}\,\overline{C}\,\overline{D}$ 和 $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$ 以得到最小项,然后将其映射;分组后的小方格如图 4.36 所示。

注意两个呈现“环绕”相邻的组。因为最外侧两列中的小方格是相邻的,所以可以形成 8 个 1 的组。由于顶部和底部的小方格是相邻的,所以形成 4 个 1 的组,用以获取剩余的两个 1。每个组的乘积项被展示出来,其结果就是最小乘积项之和表达式:

$$\overline{D} + \overline{B}C$$

记住这个最小表达式和原来的标准表达式是等价的。

相关问题： 使用卡诺图化简下面的乘积项之和表达式：

$$\overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z} + W\overline{X}YZ + W\overline{X}\,\overline{Y}Z + \overline{W}YZ + W\overline{X}\,\overline{Y}\,\overline{Z}$$

4.9.4　直接从真值表映射

前面已经看到了怎样映射布尔表达式，现在将学习怎样从真值表直接得到卡诺图。回顾一下，真值表给出了一个布尔表达式所有可能输入变量组合所对应的输出。布尔表达式的一个例子及它的真值表表示法如图 4.37 所示。注意在这个真值表中，对于 4 个不同的输入变量组合，输出 X 为 1。真值表输出列中的 1 被直接映射到卡诺图的小方格中，这些小方格对应于相关的输入变量组合的值，如图 4.37 所示。在这个图表中，可以看到布尔表达式、真值表及卡诺图仅是表示逻辑功能的不同方式而已。

$X = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C} + ABC$

输入 A	B	C	输出 X
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

AB \ C	0	1
00	1	
01		
11	1	1
10	1	

图 4.37　真值表到卡诺图的直接映射

4.9.5　“无关”条件

有时候会发生一种情况，其中有些输入变量组合是不允许出现的。例如第 2 章介绍的 BCD 码，其中有 6 个无效的组合：1010,1011,1100,1101,1110, 1111。因为这些不允许的状态在涉及 BCD 码的应用中永远不会发生，可以将其当做“无关”项来处理，因为它们对输出没有影响。也就是说，对于这些“无关”项，可以赋予输出一个 1 或者 0，这种处理没有影响，因为它们永远不会发生。

在卡诺图中，“无关”项的使用有其好处。图 4.38 在“无关”项的小方格处放置一个 X。在对 1 分组时，可以将 X 当做 1 以获取更大的组，如果方便也可以当做 0。组越大，那么结果得到的乘积项也越简单。

图 4.38(a) 中的真值表描述了一个逻辑功能，只有当 BCD 码 7、8、9 出现在输入上时，输出才会是 1。如果“无关”项被用做 1，那么该功能的结果表达式就是 $A + BCD$，如图 4.38(b) 所示。如果“无关”项未用做 1，那么结果表达式就是 $A\overline{B}\,\overline{C} + \overline{A}BCD$。所以可以看到使用“无关”项得到最简单表达式的优点。

例 4.32　在一个 7 段显示器中，7 段中每一段的点亮用于表示不同的数字组合。例如 0、2、3、5、6、7、8 和 9 的点亮，如图 4.39 所示。由于每个数字可以由 BCD 码表示，因此使用变量 $ABCD$，推导出 a 段的乘积项之和表达式。然后使用卡诺图最小化表达式。

解： a 段显示的表达式为

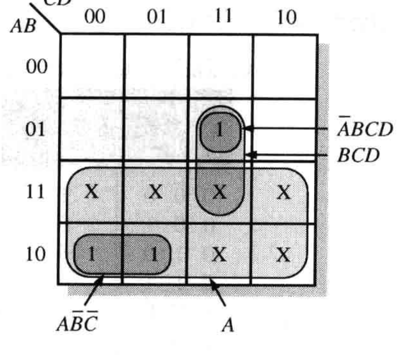
$$a = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}D$$

表达式中的每一项为使用 a 段表示的一个数字。卡诺图的最小化如图 4.40 所示。输入的 X(无关项)位表示 BCD 码中不会发生那些状态。

输入 A B C D	输出 Y
0 0 0 0	0
0 0 0 1	0
0 0 1 0	0
0 0 1 1	0
0 1 0 0	0
0 1 0 1	0
0 1 1 0	0
0 1 1 1	1
1 0 0 0	1
1 0 0 1	1
1 0 1 0	X
1 0 1 1	X
1 1 0 0	X
1 1 0 1	X
1 1 1 0	X
1 1 1 1	X

无关

(a) 真值表



(b) 没有"无关"项 $Y = A\overline{B}\,\overline{C} + \overline{A}BCD$
有"无关"项 $Y = A + BCD$

图 4.38 使用"无关"条件化简表达式

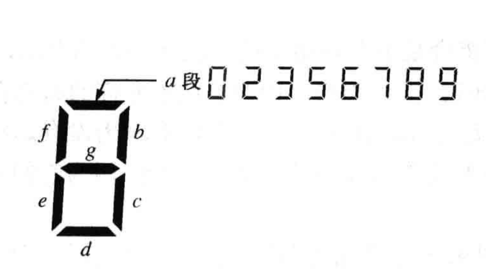


图 4.39 7 段显示器

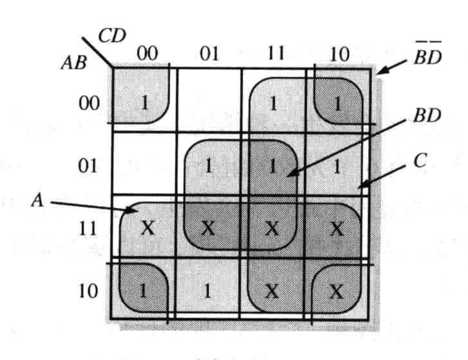


图 4.40

根据卡诺图，得到的 a 段的最小化表达式为

$$a = A + C + BD + \overline{B}\,\overline{D}$$

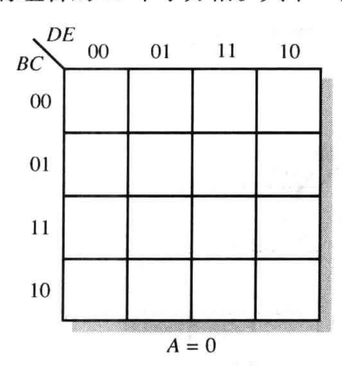
相关问题： 为 a 段逻辑画出逻辑框图。

4.9 节 温故而知新

1. 画出 3 变量和 4 变量卡诺图。
2. 为图 4.27 的卡诺图圈 1 成组，写出最小乘积项之和表达式。
3. 为图 4.34 中的每一个卡诺图，写出原来的最小项表达式。

4.10　5变量卡诺图

5变量($ABCDE$)卡诺图可以使用两个已经熟悉的4变量卡诺图来组建。每个4变量图包含了变量B、C、D、E所有组合的16个小方格。其中一张图表示$A=0$，另一张表示$A=1$，如图4.41所示。



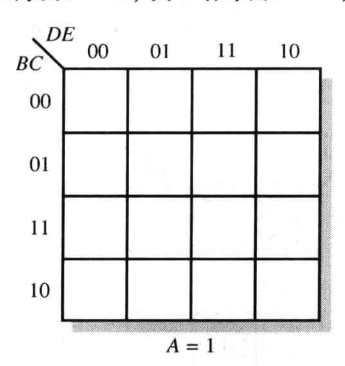


图4.41　5变量卡诺图

4.10.1　相邻项

我们已经知道如何确定4变量卡诺图的相邻项。确定16个小方格卡诺图的相邻项的直观方法是，想象$A=0$的图放置在$A=1$的图的上方。每个$A=0$的小方格与对应底下$A=1$的小方格相邻。

为了解释的目的，具有三维效果的图4.42给出了具有4个一组的例子。右侧4个为1的浅色小方格形成8位一组(4个在$A=0$的图中，4个在$A=1$的图中)。最深色小方格形成4位一组。深色小方格仅在$A=0$的图中形成4位一组。$A=1$的图中斜线小方格的1和$A=0$的图中浅色小方格的1形成2位一组。

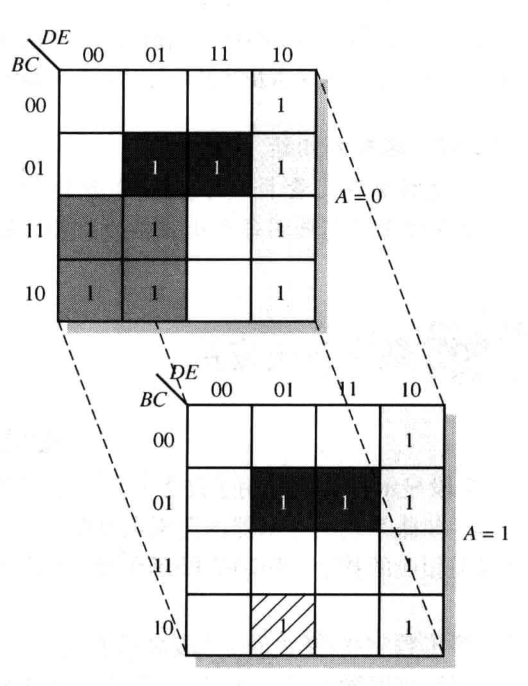


图4.42　5变量卡诺图相邻项的组合解释

确定布尔表达式　图4.42给出了最初的乘积项之和表达式的卡诺图，因为图上有17个1，所以它包含17个5变量的与项。只有组中的变量没有从反码变到原码或相反，那么表达式中才会保留此变量。从卡诺图得到简化的表达式如下：

- 浅色组的与项为$D\overline{E}$。
- 最深色组的与项为$\overline{B}CE$。
- 深色组的与项为$\overline{A}B\overline{D}$。
- 斜线和浅色组的与项为$B\overline{C}\overline{D}E$。

把这些与项加起来形成简化的乘积项之和表达式：

$$X = D\overline{E} + \overline{B}CE + \overline{A}B\overline{D} + B\overline{C}\overline{D}E$$

例 4.33 使用卡诺图化简如下的 5 变量的最小项表达式：

$$X = \bar{A}\,\bar{B}\,\bar{C}\,\bar{D}\,\bar{E} + \bar{A}\,\bar{B}C\bar{D}\,\bar{E} + \bar{A}BC\bar{D}\,\bar{E} + \bar{A}B\bar{C}\,\bar{D}\,\bar{E} + \bar{A}\,\bar{B}\,\bar{C}\,\bar{D}E + \bar{A}BC\bar{D}E$$

$$+ \bar{A}BCDE + A\bar{B}\,\bar{C}\,\bar{D}\,\bar{E} + A\bar{B}\,\bar{C}\,\bar{D}E + ABC\bar{D}E + ABCED + A\bar{B}CDE$$

解：为乘积项之和表达式画出卡诺图。图 4.43 给出了对应的与项的 1 的组合，把它们加起来形成如下简化的乘积项之和表达式：

$$X + \bar{A}\,\bar{D}\,\bar{E} + \bar{B}\,\bar{C}\,\bar{D} + BCE + ACDE$$

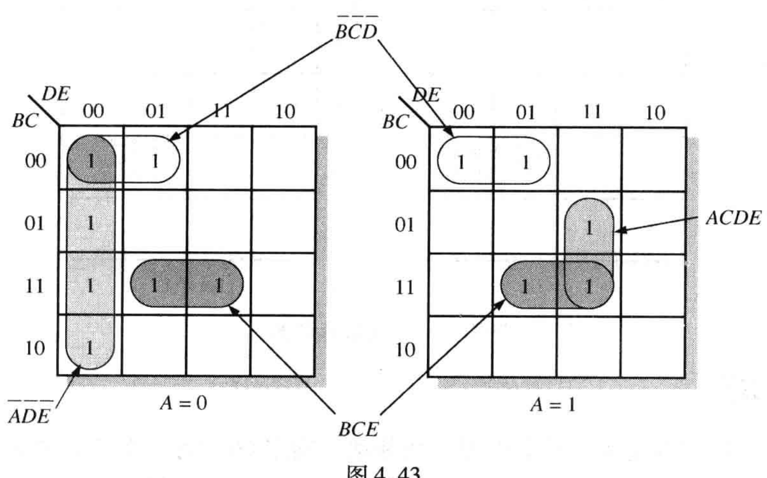


图 4.43

相关问题：化简如下的表达式：

$$Y = \bar{A}\,\bar{B}\,\bar{C}\,\bar{D}\,\bar{E} + \bar{A}\,\bar{B}C\bar{D}\,\bar{E} + \bar{A}BC\bar{D}\,\bar{E} + \bar{A}B\bar{C}\,\bar{D}\,\bar{E} + A\bar{B}\,\bar{C}\,\bar{D}\,\bar{E} + A\bar{B}C\bar{D}\,\bar{E} + ABC\bar{D}\,\bar{E} + AB\bar{C}\,\bar{D}\,\bar{E}$$
$$+ \bar{A}\,\bar{B}\,\bar{C}D\bar{E} + \bar{A}\,\bar{B}CD\bar{E} + \bar{A}BCD\bar{E} + \bar{A}B\bar{C}D\bar{E} + A\bar{B}\,\bar{C}D\bar{E} + A\bar{B}CD\bar{E} + ABCD\bar{E} + AB\bar{C}D\bar{E}$$

4.10 节 温故而知新

1. 为什么 5 变量卡诺图需要 32 个小方格？
2. 5 变量卡诺图的每个小方格为 1 时，它的表达式是什么？



数字系统应用

7 段显示器逻辑

7 段显示器可以应用于许多种日常所见的产品中。有一种药片装瓶系统就使用了 7 段显示器。装瓶系统的显示器由逻辑电路驱动，逻辑电路用二-十进制数(BCD)译码，并在显示器上点亮相应的数字。BCD-7 段译码器/驱动电路有单个的集成电路封装芯片，用于显示十进制数。

除了数字 0 ~ 9 之外，7 段显示器还可以显示某些字母。对于药片装瓶系统，需要使用十六进制键盘区输入数字和字母，使得共阳 7 段显示器分别增加字母 A、b、C、d 和 E 的显示。这些字母用来区别给定时间里装瓶的维生素药片的种类。在这个应用中，系统的译码逻辑用来显示这 5 个字母。

7 段显示器

7 段显示器的两种类型是 LED(发光二极管)和 LCD(液晶显示器)。LED 显示器中 7 段的每一段使用一个发光二极管，当电流流过时产生具有色彩的光，并且可以在黑暗环境中显示。LCD 即液晶显示器的某一段没有加电压时，由于极化的液晶反射入射光线，使得这一段与背景融为一体而不可见。然而，当某一段加电压时，液晶不反射光线，这段显示黑色。液晶显示器不能在黑暗环境中使用。

如图 4.44(a)所示为 LED 和 LCD 的 7 段的排列，标记为 a、b、c、d、e、f 和 g。选择相应的段点亮，以显示 10 个数字和某些字母，如图 4.46(b)所示。字母 b 显示小写形式，因为大写的 B 和数字 8 相同。同样，大写字母 D 的显示为 0，所以也显示小写形式。

1. 列出数字 2 的显示段。
2. 列出数字 5 的显示段。
3. 列出字母 A 的显示段。
4. 列出字母 E 的显示段。
5. 是否有所有数字共有的段?
6. 是否有所有字母共有的段?

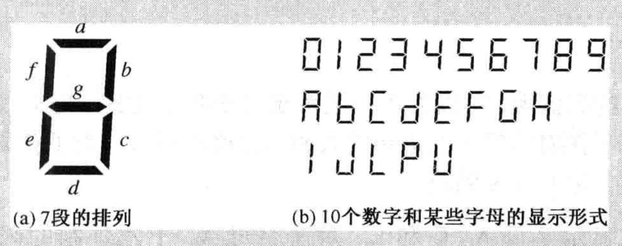


图 4.44　7 段显示器

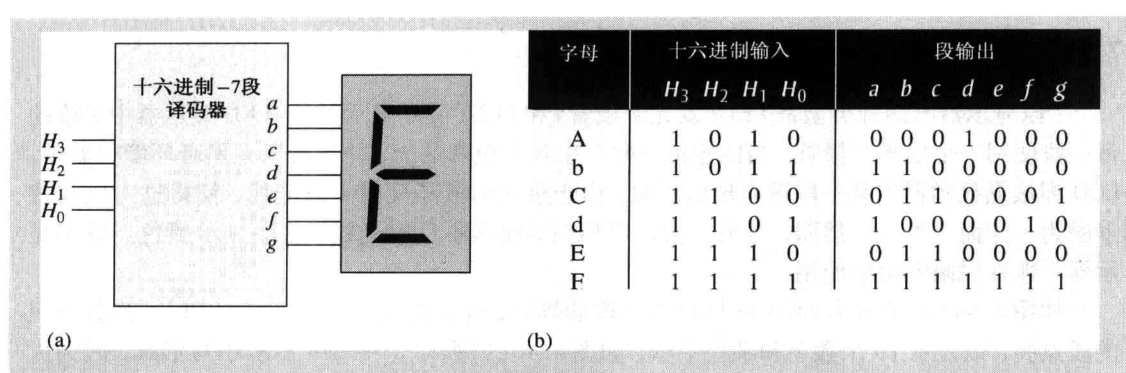
显示逻辑

表 4.9　系统显示器中 5 个字母的每一个字母所需点亮的段

字母	所需点亮的段
A	a, b, c, e, f, g
b	c, d, e, f, g
C	a, d, e, f
d	b, c, d, e, g
E	a, d, e, f, g

7 段显示器的每一段可以用来显示各种形式的字母，如图 4.44(b)所示。每一段必须由各自的译码电路点亮，用来检测任意字母所需要的段码。因为使用共阳显示器，所以各段由低电平 0 点亮，由高电平 1 熄灭。表 4.9 给出了装瓶系统中每个字母需要点亮的段。尽管由低电平点亮(点亮 LED)，但是其他情况的逻辑表达式与本章讨论的完全相同。通过映射每个可能的输入对应所需的输出(1、0 或 X)，在卡诺图中圈 1 成组，然后从图中获取乘积项之和表达式。同样，可以让给出的段熄灭以减少逻辑表达式。开始时，这个方法似乎有些混乱，但用过以后就简单了，它避免了双极型(TTL)逻辑电路的输出电流的限制问题(第 12 章将讨论)。

图 4.45(a)为产生 5 个字母的 7 段逻辑和显示的框图，图 4.45(b)为其真值表。逻辑电路框图包括 4 个十六进制输入和 7 个输出，每个输出对应于一个段。因为输入中没有使用字母 F，真值表中对应的输出全部置 1(熄灭)。



字母	十六进制输入				段输出						
	H_3	H_2	H_1	H_0	a	b	c	d	e	f	g
A	1	0	1	0	0	0	0	1	0	0	0
b	1	0	1	1	1	1	0	0	0	0	0
C	1	1	0	0	0	1	1	0	0	0	1
d	1	1	0	1	1	0	0	0	0	1	0
E	1	1	1	0	0	1	1	0	0	0	0
F	1	1	1	1	1	1	1	1	1	1	1

图 4.45　使用的系统中字母 A ~ F 的十六进制 - 7 段译码器

卡诺图和无效 BCD 码的检测

将图 4.45 的真值表映射到卡诺图，为每一段开发一个简单的逻辑电路。对于字母显示中不显示的 BCD 码，在卡诺图中将用 X 表示(无关)。这会使得逻辑表达更简洁，但是会导致某些奇怪的输出，除非采取措施不让这种情况发生。当在键盘区输入无效码时，由于仅让这些无效的 BCD 码有输出，因此就只显示这些无效码字母。

段逻辑的表达式

使用图 4.45(b)的真值表，可以为每个段写出最小项表达式，然后使用卡诺图化简。真值表所需的输出在卡诺图对应的十六进制输入的小方格中写入。为了获取显示逻辑所需的最小乘积项之和表达式，对 1 和 X 圈组。

a 段　a 段用来显示字母 A、C 和 E。对于字母 A，十六进制的码为 1010，或以变量 $H_3\overline{H}_2H_1\overline{H}_0$ 的形式表示。对于字母 C，十六进制的码为 1100，或以变量 $H_3H_2\overline{H}_1\overline{H}_0$ 的形式表示。对于字母 E，十六进制的码为 1110，或以变量 $H_3H_2H_1\overline{H}_0$ 的形式表示。这样 a 段的最小项表达式为

$$a = H_3\overline{H}_2H_1\overline{H}_0 + H_3H_2\overline{H}_1\overline{H}_0 + H_3H_2H_1\overline{H}_0$$

由于低电平为每个段逻辑电路的有效输出状态，因此根据字母应显示的段所表示的码，在对应的卡诺图的每个小方格中写入 0。完成 1 和 X 圈组以后，a 段对应的简化表达式如图 4.46(a)所示。

b 段　b 段用来显示字母 A 和 d。b 段的最小项表达式为

$$b = H_3\overline{H}_2H_1\overline{H}_0 + H_3H_2\overline{H}_1H_0$$

b 段对应的简化表达式如图 4.46(b)所示。

c 段　c 段用来显示字母 A、b 和 d。c 段的最小项表达式为

$$c = H_3\overline{H}_2H_1\overline{H}_0 + H_3\overline{H}_2H_1H_0 + H_3H_2\overline{H}_1H_0$$

c 段对应的简化表达式如图 4.46(c)所示。

7. 为 d 段开发简化表达式。

8. 为 e 段开发简化表达式。
9. 为 f 段开发简化表达式。
10. 为 g 段开发简化表达式。

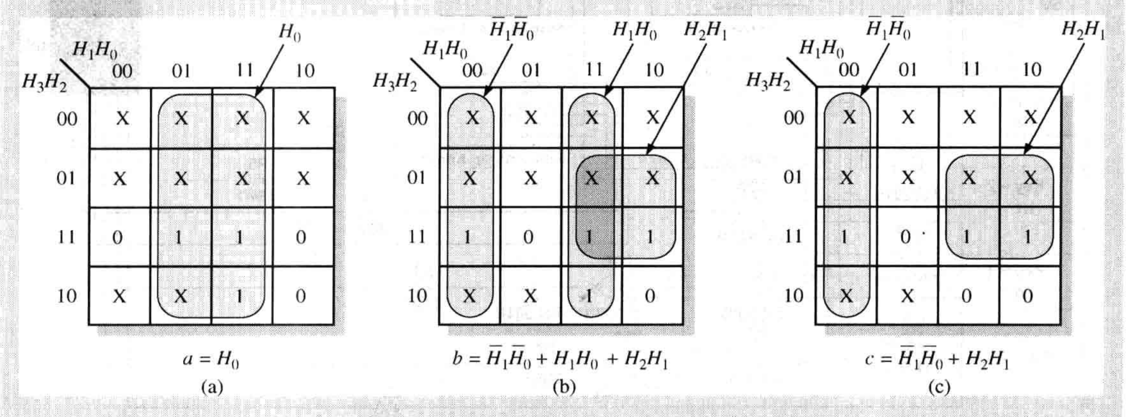


图 4.46　a 段、b 段、c 段表达式的最小化

逻辑电路

根据最小化的表达式，可以得到每一段的逻辑电路。对于 a 段，直接把输入 H_0 连接到显示的 a 段上(不需要门电路)。如图 4.47 所示，对于 b 段和 c 段使用与门或者或门。注意表达式中 b 段和 c 段都具有的与项(H_2H_1 和 $\overline{H}_1\ \overline{H}_0$)，因此如给出的电路图，它们都可以使用与门来实现。

11. 给出 d 段的逻辑电路。
12. 给出 e 段的逻辑电路。
13. 给出 f 段的逻辑电路。
14. 给出 g 段的逻辑电路。

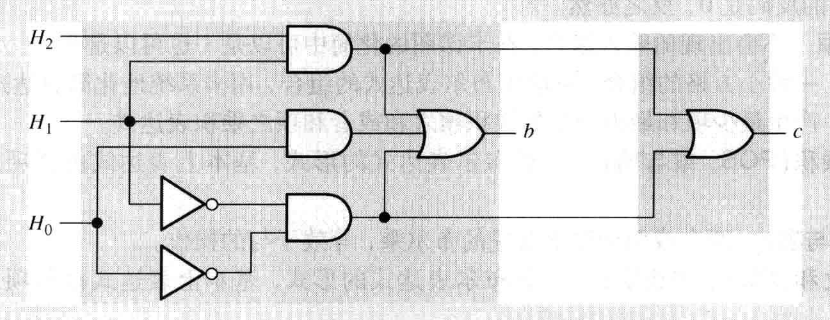


图 4.47　b 段和 c 段的逻辑电路

电路仿真

如图 4.48 所示，使用 Multisim 实现所选字母 E 的译码器仿真。在课外或实验室里实现用于段逻辑的子电路。

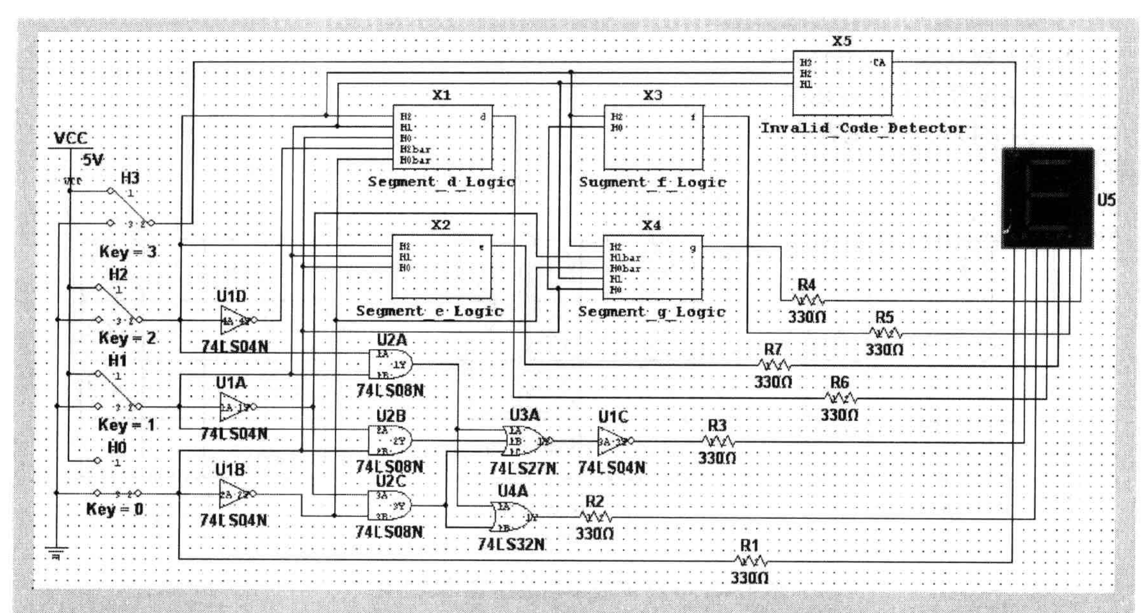


图 4.48 译码器和显示器的 Multisim 仿真电路的界面

打开网络资源中系统应用示例的文件 SAA04。使用 Multisim 软件运行译码器和显示器仿真文件，观察指定字母的运行结果。

把知识用于实践

如何修改译码器用于 7 段共阴显示器。

关键词

求反 求一个数的反码或得到相反的结果。在布尔代数中，求反的功能是在一个变量上加一个横杠。1 的反码是 0，反之亦然。

"无关"项 不会出现的输入组合，在卡诺图的化简中可以是 1 也可以是 0。

卡诺图 一种小方格的组合，对应于布尔表达式的组合，用来系统地化简表达式。

最小化 产生最少项和最少变量的乘积项之和或者和项之乘积表达式。

和项之乘积(POS，或与项) 一种布尔表达式的形式，基本上表达式由或项之间相与而形成。

乘积项(与项) 两个或两个以上变量的布尔乘，等效于与的操作。

乘积项之和(SOP，与或项) 一种布尔表达式的形式，基本上表达式由与项之间相或而形成。

和项(或项) 两个或两个以上变量的布尔加，等效于或的操作。

变量 一个符号，用来表示一种作用、状态或数据，可以有 1 或 0 两个值，通常由斜体字母或单词来表示。

判断题 (答案在本章的结尾。)

1. 变量、反码和字母在布尔代数中都会使用。

2. 在布尔代数中的加等效于或的运算。
3. 在布尔代数中的乘等效于与非的运算。
4. 交换律、结合律和分配律都是布尔代数中的定律。
5. 1 的反码是 0。
6. 当一个布尔变量乘以它的反码时，结果就是此变量。
7. “变量的乘积项等于每个变量取反以后相或”是狄摩根定理的一种陈述。
8. 与或项指的是一系列乘积。
9. 卡诺图可以用来化简布尔表达式。
10. 一个 4 变量的卡诺图有 8 个小方格。

自测题 （答案在本章的结尾。）

1. 变量的反码始终是
   (a)0　(b)1　(c)等于变量　(d)变量的反相
2. 布尔表达式 $A+\overline{B}+C$ 是
   (a)一个和项　(b)一个文字项　(c)一个乘积项　(d)一个补码项
3. 布尔表达式 $A\overline{B}C\overline{D}$ 是
   (a)一个和项　(b)一个乘积项　(c)一个文字项　(d)始终是 1
4. 表达式 $A\overline{B}CD+A\overline{B}+\overline{C}D+B$ 的域是
   (a)A 和 D　(b)只是 B　(c)A、B、C 和 D　(d)都不是
5. 按照加法的交换律，
   (a)$AB=BA$　(b)$A=A+A$
   (c)$A+(B+C)=(A+B)+C$　(d)$A+B=B+A$
6. 按照乘法的结合律，
   (a)$B=BB$　(b)$A(BC)=(AB)C$
   (c)$A+B=B+A$　(d)$B+B=(B+0)$
7. 按照乘法的分配律，
   (a)$A(B+C)=AB+AC$　(b)$A(BC)=ABC$
   (c)$A(A+1)=A$　(d)$A+AB=A$
8. 下面哪一个不是布尔代数的有效法则？
   (a)$A+1=1$　(b)$A=\overline{A}$　(c)$AA=A$　(d)$A+0=A$
9. 下面哪一个法则表述了如果与门的一个输入总是 1，那么输出就等于另外一个输入？
   (a)$A+1=1$　(b)$A+A=A$　(c)$A\cdot A=A$　(d)$A\cdot 1=A$
10. 按照狄摩根定理，下面的等式是正确的有
   (a) $\overline{AB}=\overline{A}+\overline{B}$　(b) $\overline{XYZ}=\overline{X}+\overline{Y}+\overline{Z}$
   (c) $\overline{A+B+C}=\overline{A}\,\overline{B}\,\overline{C}$　(d)所以这些等式
11. 布尔表达式 $X=AB+CD$ 表示
   (a)两个或运算再相与　(b)一个 4 输入与门
   (c)两个与运算再相或　(d)一个异或门
12. 一个乘积项之和表达式的例子是
   (a)$A+B(C+D)$　(b) $\overline{A}B+A\overline{C}+A\overline{B}C$　(c)$(\overline{A}+B+C)(A+\overline{B}+C)$　(d)答案(a)和(b)
13. 一个和项之乘积表达式的例子是
   (a)$A(B+C)+A\overline{C}$　(b)$(A+B)(\overline{A}+B+\overline{C})$　(c)$\overline{A}+\overline{B}+BC$　(d)答案(a)和(b)
14. 一个最小项之和表达式的例子是

(a) $\overline{A}B + A\overline{B}C + AB\overline{D}$　　(b) $A\overline{B}C + A\overline{C}D$

(c) $A\overline{B} + \overline{A}B + AB$　　(d) $A\overline{B}C\overline{D} + \overline{A}B + \overline{A}$

15. 3 变量卡诺图有

(a)8 个小方格　(b)3 个小方格　(c)16 个小方格　(d)4 个小方格

16. 在 4 变量卡诺图中，一个 2 变量乘积项是由哪一项产生的?

(a)1 的 2 个小方格组　　(b)1 的 8 个小方格组

(c)1 的 4 个小方格组　　(d)0 的 4 个小方格组

17. 5 变量的卡诺图有

(a)16 个小方格　(b)32 个小方格　(c)64 个小方格

18. 奎恩-麦克拉斯基化简法(Q-M 法)可以用来

(a)替代卡诺图的方法　　(b)简化 5 个或 5 个以上变量的表达式

(c)答案(a)和(b)　　(d)都不是

习题 (奇数题的答案在本书的结尾。)

4.1 节　布尔运算和表达式

1. 使用布尔符号，写出一个表达式，无论何时它的一个或多个变量(A、B、C、D)为 1 时，这个表达式为 1。
2. 写出一个表达式，当且仅当所有的变量(A、B、C、D 和 E)为 1 时，这个表达式才是 1。
3. 写出一个表达式，当一个或多个变量(A、B 和 C)为 1 时，这个表达式才是 0。
4. 计算下面的运算：

(a)$0+0+1$　(b)$1+1+1$　(c)$1\cdot 0\cdot 0$　(d)$1\cdot 1\cdot 1$

(e)$1\cdot 0\cdot 1$　(f)$1\cdot 1+0\cdot 1\cdot 1$

5. 求使得乘积项为 1、和项为 0 的变量的值：

(a)AB　(b)$A\overline{B}C$　(c)$A+B$　(d)$\overline{A}+B+\overline{C}$

(e)$\overline{A}+\overline{B}+C$　(f)$\overline{A}+B$　(g)$A\overline{B}\overline{C}$

6. 对于变量的所有可能的值，求 X 的值。

(a)$X=(A+B)C+B$　(b)$X=(\overline{A+B})C$　(c)$X=A\overline{B}C+AB$

(d)$X=(A+B)(\overline{A}+B)$　(e)$X=(A+BC)(\overline{B}+\overline{C})$

4.2 节　布尔代数的定律和法则

7. 基于下面的每一个等式，指出布尔代数的定律：

(a) $A\overline{B} + CD + A\overline{C}D + B = B + A\overline{B} + A\overline{C}D + CD$

(b) $AB\overline{C}D + \overline{ABC} = D\overline{C}BA + \overline{CBA}$

(c) $AB(CD + E\overline{F} + GH) = ABCD + ABE\overline{F} + ABGH$

8. 基于下面的每一个等式，指出布尔代数的定律：

(a) $\overline{\overline{AB + CD}} + \overline{EF} = AB + CD + \overline{EF}$　　(b) $A\overline{A}B + AB\overline{C} + AB\overline{B} = AB\overline{C}$

(c)$A(BC+BC)+AC=A(BC)+AC$　　(d)$AB(C+\overline{C})+AC=AB+AC$

(e)$A\overline{B}+A\overline{B}C=A\overline{B}$　　(f)$ABC+\overline{AB}+\overline{ABC}D=ABC+\overline{AB}+D$

4.3 节　狄摩根定理

9. 对于下面的每一个表达式，应用狄摩根定理：

(a) $\overline{A+\overline{B}}$　(b) $\overline{\overline{A}B}$　(c) $\overline{A+B+C}$　(d) $\overline{ABC}$

(e) $\overline{A(B+C)}$　(f) $\overline{AB}+\overline{CD}$　(g) $\overline{AB+CD}$　(h) $\overline{(A+\overline{B})(\overline{C}+D)}$

10. 对于下面的每一个表达式，应用狄摩根定理：

(a) $\overline{A\overline{B}(C+\overline{D})}$　　(b) $\overline{AB(CD+EF)}$　　(c) $\overline{(A+\overline{B}+C+\overline{D})}+\overline{ABC\overline{D}}$

(d) $\overline{\overline{(\overline{A}+B+C+D)}(A\overline{B}\overline{C}D)}$　　(e) $\overline{\overline{A}\overline{B}(CD+\overline{E}F)(\overline{AB}+\overline{CD})}$

11. 对于下面的每一个表达式，应用狄摩根定理：

(a) $\overline{\overline{(\overline{ABC})(\overline{EFG})}+\overline{(\overline{HIJ})(\overline{KLM})}}$　　(b) $\overline{(A+\overline{B}\overline{\overline{C}}+CD)}+\overline{\overline{BC}}$

(c) $\overline{\overline{\overline{(\overline{A}+B)}\,\overline{(C+D)}\,\overline{(E+F)}\,\overline{(G+H)}}}$

4.4 节　逻辑电路的布尔分析

12. 在图4.49中，为每一个逻辑门写出布尔表达式。

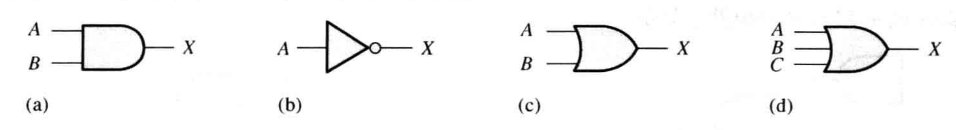


图4.49

13. 在图4.50中，为每一个逻辑门写出布尔表达式。

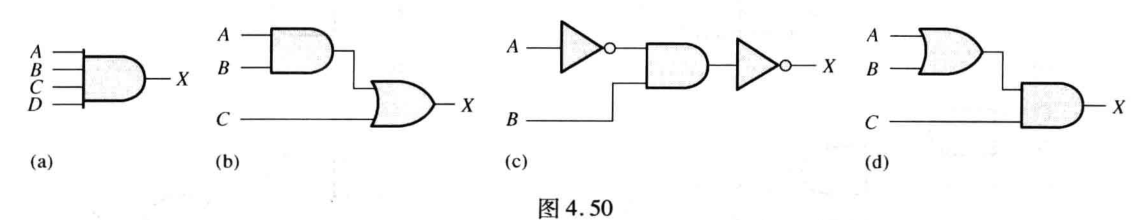


图4.50

14. 绘制下面每一个表达式的逻辑电路：

(a) $A+B+C$　　(b) ABC　　(c) $AB+C$　　(d) $AB+CD$

15. 绘制下面每一个表达式的逻辑电路：

(a) $A\overline{B}+\overline{A}B$　　(b) $AB+\overline{AB}+\overline{A}BC$

(c) $\overline{A}B(C+\overline{D})$　　(d) $A+B[C+D(B+\overline{C})]$

16. (a) 为如下输出的情况画出逻辑电路图，仅当ASSERT和READY两者为高电平时，使能(ENABLE)为低电平。

(b) 为如下输出的情况画出逻辑电路图，仅当LOAD为高电平并且输入READY为低电平时，输出保持(HOLD)为低电平。

17. 为图4.51的每个电路图写出真值表。

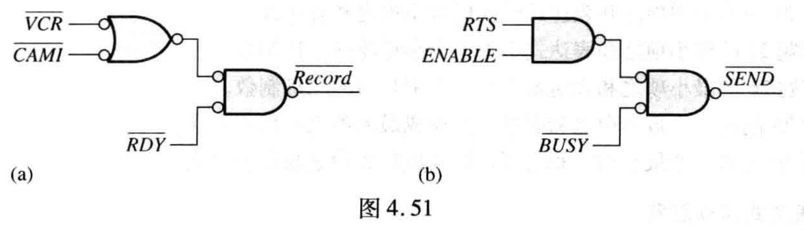


图4.51

18. 为下面的每一个布尔表达式构建真值表：

(a) $A+B$　　(b) AB　　(c) $AB+BC$　　(d) $(A+B)C$

(e) $(A+B)(\overline{B}+C)$

4.5 节　使用布尔代数进行化简

19. 使用布尔代数，化简下面的表达式：

(a) $A(A+B)$　　(b) $A(\overline{A}+AB)$　　(c) $BC+\overline{B}C$

(d) $A(A+\overline{A}B)$　　(e) $A\overline{B}C+\overline{A}BC+\overline{A}\overline{B}C$

20. 使用布尔代数，化简下面的表达式：

(a) $(A+\overline{B})(A+C)$ (b) $\overline{A}B+\overline{A}B\overline{C}+\overline{A}BCD+\overline{A}B\overline{C}\overline{D}E$

(c) $AB+\overline{AB}C+A$ (d) $(A+\overline{A})(AB+AB\overline{C})$

(e) $AB+(\overline{A}+\overline{B})C+AB$

21. 使用布尔代数，化简下面的表达式：

(a) $BD+B(D+E)+\overline{D}(D+F)$ (b) $\overline{A}\,\overline{B}C+\overline{(A+B+\overline{C})}+\overline{A}\,\overline{B}\overline{C}D$

(c) $(B+BC)(B+\overline{B}C)(B+D)$ (d) $ABCD+AB(\overline{CD})+(\overline{AB})CD$

(e) $ABC[AB+\overline{C}(BC+AC)]$

22. 确定图 4.52 中哪些电路是等价的。

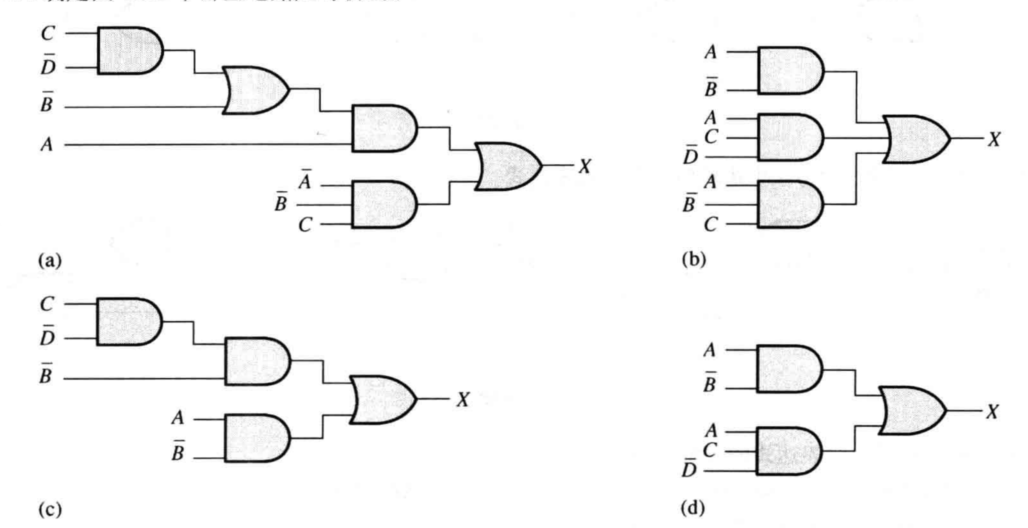


图 4.52

4.6 节 布尔表达式的标准形式

23. 把下面的表达式转换成乘积项之和的形式：

(a) $(A+B)(C+\overline{B})$ (b) $(A+\overline{B}C)C$ (c) $(A+C)(AB+AC)$

24. 把下面的表达式转换成乘积项之和的形式：

(a) $AB+CD(A\overline{B}+CD)$ (b) $AB(\overline{B}\overline{C}+BD)$ (c) $A+B[AC+(B+\overline{C})D]$

25. 确定习题 23 中的每一个乘积项之和表达式的域，并且把它们转换成最小项之和表达式。

26. 把习题 24 中的乘积项之和表达式转换成最小项之和表达式。

27. 确定习题 25 的最小项之和表达式中每一个乘积项的二进制数。

28. 确定习题 26 的最小项之和表达式中每一个乘积项的二进制数。

29. 把习题 25 的每一个最小项之和表达式转换成最大项之乘积表达式。

30. 把习题 26 的每一个最小项之和表达式转换成最大项之乘积表达式。

4.7 节 布尔表达式和真值表

31. 为下面每一个最小项之和表达式开发真值表：

(a) $A\overline{B}C+\overline{A}B\overline{C}+ABC$

(b) $\overline{X}\,\overline{Y}\,\overline{Z}+\overline{X}\overline{Y}Z+XY\overline{Z}+X\overline{Y}Z+\overline{X}YZ$

32. 为下面每一个最小项之和表达式开发真值表：

(a) $\overline{A}B\overline{C}D+\overline{A}BC\overline{D}+A\overline{B}\overline{C}D+\overline{A}\,\overline{B}\overline{C}\overline{D}$

(b) $WXYZ+WXY\overline{Z}+\overline{W}XYZ+W\overline{X}YZ+WX\overline{Y}Z$

33. 为下面每一个乘积项之和表达式开发真值表：

(a) $\overline{A}B+AB\overline{C}+\overline{A}\,\overline{C}+A\overline{B}C$

(b) $\bar{X} + Y\bar{Z} + WZ + X\bar{Y}Z$

34. 为下面每一个最大项之乘积表达式开发真值表：

(a) $(\bar{A} + \bar{B} + \bar{C})(A + B + C)(A + \bar{B} + C)$

(b) $(\bar{A} + B + \bar{C} + D)(A + \bar{B} + C + \bar{D})(A + \bar{B} + \bar{C} + D)(\bar{A} + B + C + \bar{D})$

35. 为下面每一个最大项之乘积表达式开发真值表：

(a) $(A + B)(A + C)(A + B + C)$

(b) $(A + \bar{B})(A + \bar{B} + \bar{C})(B + C + \bar{D})(\bar{A} + B + \bar{C} + D)$

36. 为表 4.10 中的每一个真值表，推导出一个最小项之和表达式和最大项之乘积表达式。

表 4.10

(a)

ABC	X
000	0
001	1
010	0
011	0
100	1
101	1
110	0
111	1

(b)

ABC	X
000	0
001	0
010	0
011	0
100	0
101	1
110	1
111	1

(c)

$ABCD$	X
0000	1
0001	1
0010	0
0011	1
0100	0
0101	1
0110	1
0111	0
1000	0
1001	1
1010	0
1011	0
1100	1
1101	0
1110	0
1111	0

(d)

$ABCD$	X
0000	0
0001	0
0010	1
0011	0
0100	1
0101	1
0110	0
0111	1
1000	0
1001	0
1010	0
1011	1
1100	1
1101	0
1110	0
1111	1

4.8 节　卡诺图

37. 绘制一个 3 变量卡诺图，并根据二进制数标记每一个小方格。

38. 绘制一个 4 变量卡诺图，并根据二进制数标记每一个小方格。

39. 在一个 3 变量卡诺图中，写出每个小方格的最小项。

4.9 节　卡诺图乘积项之和的最小化

40. 使用卡诺图为每一个表达式求出最小乘积项之和的形式：

(a) $\bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}C$　　(b) $AC(\bar{B} + C)$

(c) $\bar{A}(BC + B\bar{C}) + A(BC + B\bar{C})$　　(d) $\bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}B\bar{C} + AB\bar{C}$

41. 使用卡诺图化简每一个表达式，以得到最小乘积项之和的形式：

(a) $\bar{A}\bar{B}\bar{C} + A\bar{B}C + \bar{A}BC + AB\bar{C}$　　(b) $AC[\bar{B} + B(B + \bar{C})]$

(c) $DE\bar{F} + \bar{D}E\bar{F} + \bar{D}\bar{E}\bar{F}$

42. 把表达式扩展成最小项之和的形式：

(a) $AB + A\bar{B}C + ABC$　　(b) $A + BC$

(c) $A\bar{B}\bar{C}D + AC\bar{D} + B\bar{C}D + \bar{A}BC\bar{D}$　　(d) $A\bar{B} + A\bar{B}\bar{C}D + CD + B\bar{C}D + ABCD$

43. 使用卡诺图对习题 42 的每个表达式进行最小化。

44. 使用卡诺图把每个表达式化简为最小乘积项之和形式：

(a) $A+B\overline{C}+CD$

(b) $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}+\overline{A}\,\overline{B}\,\overline{C}D+ABCD+ABC\overline{D}$

(c) $\overline{A}B(\overline{C}\,\overline{D}+\overline{C}D)+AB(\overline{C}\,\overline{D}+\overline{C}D)+A\overline{B}\,\overline{C}D$

(d) $(\overline{A}\,\overline{B}+A\overline{B})(CD+C\overline{D})$

(e) $\overline{A}\,\overline{B}+A\overline{B}+\overline{C}\,\overline{D}+C\overline{D}$

45. 使用卡诺图把表 4.11 所确定的逻辑函数化简为最小乘积项之和的形式。

46. 使用卡诺图把表 4.12 所确定的逻辑函数化简为最小乘积项之和的形式。

表 4.11

输入			输出
A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

表 4.12

输入				输出
A	B	C	D	X
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

47. 重做习题 46，把最后 6 个二进制数组合为不允许的项。

4.10 节 5 变量卡诺图

48. 画出表达式 $A\overline{B}C\overline{D}E+\overline{A}\,\overline{B}C\overline{D}E$ 的卡诺图，并且尽可能地化简。

49. 使用卡诺图最小化如下的乘积项之和表达式。

$$X=\overline{A}B\overline{C}D\overline{E}+\overline{A}\,\overline{B}\,\overline{C}DE+A\overline{B}\,\overline{C}DE+AB\overline{C}\,\overline{D}\,\overline{E}+\overline{A}BCD\overline{E}+\overline{A}BC\overline{D}E$$
$$+\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\,\overline{E}+\overline{A}\,\overline{B}CDE+AB\overline{C}D\overline{E}+AB\overline{C}DE$$

50. 使用卡诺图最小化如下的乘积项之和表达式。

$$A=\overline{V}WXYZ+V\overline{W}XYZ+VW\overline{X}YZ+VWX\overline{Y}Z+VWXY\overline{Z}+\overline{V}\,\overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z}$$
$$+\overline{V}\,\overline{W}\,\overline{X}Y\overline{Z}+\overline{V}\,\overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z}+\overline{V}W\overline{X}\,\overline{Y}\,\overline{Z}$$

数字系统应用

51. 如果需要选择一个在黑暗环境下工作的同一种数值显示器，选择 LED 类型还是 LCD 类型的 7 段显示器？为什么？

52. 解释无效码检测器的用途。

53. 对于 c 段，实现最小乘积项之和表达式所需的最少的门和反相器，比最小项之和表达式所需的要少多少？

54. 对于 d 段到 g 段，重复习题 53。

答案

温故而知新

4.1 节　布尔运算和表达式

1. $\overline{A}=\overline{0}=1$

2. $A=1$，$B=1$，$C=0$；$\overline{A}+\overline{B}+C=\overline{1}+\overline{1}+0=0+0+0=0$

3. $A=1$，$B=0$，$C=1$；$A\overline{B}C=1\cdot\overline{0}\cdot 1=1\cdot 1\cdot 1=1$

4.2 节　布尔代数的定律和法则

1. $A+(B+C+D)=(A+B+C)+D$　　2. $A(B+C+D)=AB+AC+AD$

4.3 节　狄摩根定理

1. (a) $\overline{ABC}+(\overline{\overline{D}+E})=\overline{A}+\overline{B}+\overline{C}+D\overline{E}$　　(b) $\overline{(A+B)C}=\overline{A}\,\overline{B}+\overline{C}$　　(c) $\overline{A+B+C}+\overline{\overline{D}E}=\overline{A}\,\overline{B}\,\overline{C}+D+\overline{E}$

4.4 节　逻辑电路的布尔分析

1. $(C+D)B+A$

2. 写出真值表，当 A 为 1 或 B 和 C 为 1，或 B 和 D 为 1 时，表达式为 1；其他的变量组合时，表达式为 0。

4.5 节　使用布尔代数进行化简

1. (a) $A+AB+A\overline{B}C=A$　　(b) $(\overline{A}+B)C+ABC=C(\overline{A}+B)$
   (c) $A\overline{B}C(BD+CDE)+A\overline{C}=A(\overline{C}+\overline{B}DE)$

2. (a) 初始电路：2 个与门，1 个或门，1 个反相器；化简结果：没有门(直接连接)。
   (b) 初始电路：2 个或门，2 个与门，1 个反相器；化简结果：1 个或门，2 个反相器。
   (c) 初始电路：5 个与门，2 个或门，1 个反相器；化简结果：2 个与门，1 个或门，2 个反相器。

4.6 节　布尔表达式的标准形式

1. (a) 与或项　　(b) 最大项　　(c) 最小项　　(d) 或与项

2. (a) $AB\overline{C}\overline{D}+AB\overline{C}D+ABC\overline{D}+ABCD+\overline{A}B\overline{C}D+\overline{A}BCD+\overline{A}\,\overline{B}C\overline{D}+\overline{A}BC\overline{D}$
   (c) 已经是标准形式。

3. (b) 已经是标准形式。
   (d) $(A+\overline{B}+\overline{C})(A+\overline{B}+C)(A+B+\overline{C})(A+B+C)$

4.7 节　布尔表达式和真值表

1. $2^5=32$　　2. $0110\longrightarrow\overline{W}XY\overline{Z}$　　3. $1100\longrightarrow\overline{W}+\overline{X}+Y+Z$

4.8 节　卡诺图

1. (a) 左上角小方格：000　　(b) 右下角小方格：101
   (c) 左下角小方格：100　　(d) 右上角小方格：001

2. (a) 左上角小方格：$\overline{X}\,\overline{Y}\,\overline{Z}$　　(b) 右下角小方格：$X\overline{Y}Z$
   (c) 左下角小方格：$X\overline{Y}\,\overline{Z}$　　(d) 右上角小方格：$\overline{X}\,\overline{Y}Z$

3. (a) 左上角小方格：0000　　(b) 右下角小方格：1010
   (c) 左下角小方格：1000　　(d) 右上角小方格：0010

4. (a) 左上角小方格：$\overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z}$　　(b) 右下角小方格：$W\overline{X}Y\overline{Z}$

(c)左下角小方格：$W\overline{X}\overline{Y}\overline{Z}$　　(d)右上角小方格：$\overline{W}\overline{X}Y\overline{Z}$

4.9 节　卡诺图乘积项之和的最小化

1. 3 变量使用 8 个小方格，4 变量使用 16 个小方格。
2. $AB+B\overline{C}+\overline{A}\,\overline{B}C$
3. (a) $\overline{A}\,\overline{B}\,\overline{C}+\overline{A}BC+ABC+AB\overline{C}$

   (b) $\overline{A}\,\overline{B}\,\overline{C}+\overline{A}\,\overline{B}C+\overline{A}B\overline{C}+ABC+A\overline{B}\,\overline{C}+A\overline{B}C$

   (c) $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}+\overline{A}\,\overline{B}\,\overline{C}D+\overline{A}B\overline{C}\,\overline{D}+\overline{A}B\overline{C}D+\overline{A}BCD+\overline{A}BC\overline{D}+A\overline{B}\,\overline{C}D+A\overline{B}CD$

   (d) $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}+\overline{A}B\overline{C}\,\overline{D}+AB\overline{C}\,\overline{D}+A\overline{B}\,\overline{C}\,\overline{D}+\overline{A}B\overline{C}D+AB\overline{C}D+A\overline{B}CD+\overline{A}\,\overline{B}C\overline{D}+\overline{A}BC\overline{D}+ABC\overline{D}+A\overline{B}C\overline{D}$

4.10 节　5 变量卡诺图

1. 5 个变量有 32 种组合（$2^5=32$）。
2. 所有 5 个变量的组合时函数为 1，所以 $X=1$ 。

例题的相关问题

4.1　当 $A=1$ 和 $B=0$ 时，$\overline{A}+B=0$ 。　　4.2　当 $A=0$ 和 $B=0$ 时，$\overline{A}\,\overline{B}=1$。

4.3　XYZ　　4.4　$W+X+Y+Z$　　4.5　$ABC\overline{D}\,\overline{E}$　　4.6　$(A+\overline{B}+\overline{C}D)\overline{E}$

4.7　$\overline{ABCD}=\overline{A}+\overline{B}+\overline{C}+\overline{D}$　　4.8　结果和例题相同。　　4.9　$A\overline{B}$　　4.10　CD

4.11　$AB\overline{C}+\overline{A}C+\overline{A}\,\overline{B}$　　4.12　$\overline{A}+\overline{B}+\overline{C}$　　4.13　结果和例题相同。

4.14　$\overline{A}B\overline{C}+AB+A\overline{C}+A\overline{B}+\overline{B}\,\overline{C}$　　4.15　$W\overline{X}YZ+W\overline{X}Y\overline{Z}+W\overline{X}Y\overline{Z}+\overline{W}\overline{X}Y\overline{Z}+WX\overline{Y}Z+WX\overline{Y}\,\overline{Z}$

4.16　011，101，110，010，111，是。　　4.17　$(A+\overline{B}+C)(A+\overline{B}+\overline{C})(A+B+C)(\overline{A}+B+C)$

4.18　010，100，001，111，011，是。　　4.19　与或项和或与项等价。

4.20　参见表 4.13。　　4.21　参见表 4.14。

表 4.13

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

表 4.14

A	B	C	X
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

4.22　与或项和或与项等价。　　4.23　参见图 4.53。　　4.24　参见图 4.54。

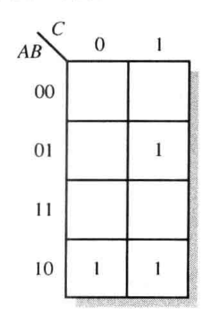


图 4.53

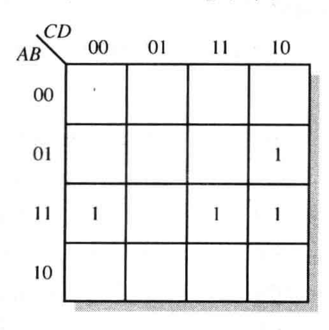


图 4.54

4.25　参见图 4.55。　4.26　参见图 4.56。

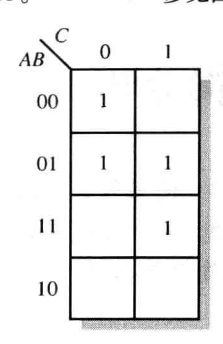


图 4.55

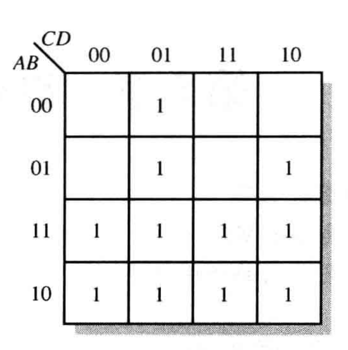


图 4.56

4.27　没有其他方法。　4.28　$X = B + \overline{A}C + A\overline{C}D + C\overline{D}$

4.29　$X = \overline{D} + A\overline{B}C + B\overline{C} + \overline{A}B$　4.30　$Q = X + Y$

4.31　$Q = \overline{X}\,\overline{Y}\,\overline{Z} + W\overline{X}Z + \overline{W}YZ$　4.32　参见图 4.57。

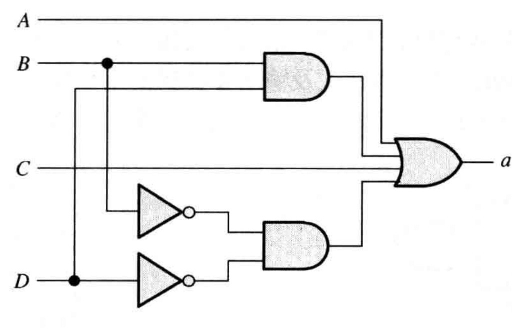


图 4.57

4.33　$Y = \overline{D}\,\overline{E} + \overline{A}\,\overline{E} + \overline{B}\,\overline{C}\,\overline{E}$

判断题

1. T　2. T　3. F　4. T　5. T　6. F　7. T　8. F　9. T　10. F

自测题

1. (d)　2. (a)　3. (b)　4. (c)　5. (d)　6. (b)　7. (a)　8. (b)
9. (d)　10. (d)　11. (c)　12. (b)　13. (b)　14. (c)　15. (a)　16. (c)
17. (b)　18. (c)

第5章　组合逻辑分析

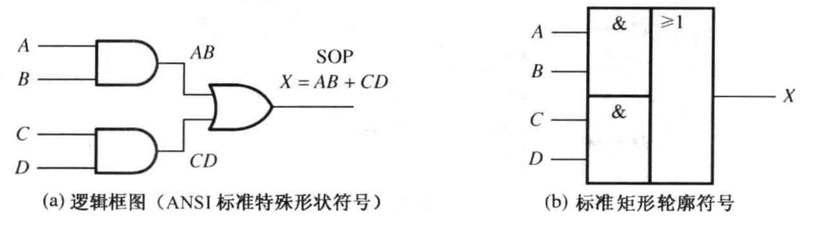
章节提纲

5.1　基本组合逻辑电路

5.1.1　与-或逻辑

◇ 与-或逻辑产生一个乘积项之和表达式。

图5.1(a)展示了一个与-或电路，由两个2输入与门和一个2输入或门组成；图5.1(b)是ANSI标准矩形轮廓符号。与门输出的布尔表达式及输出 X 得到的乘积项之和表达式如框图所示。一般来说，一个与-或电路可以具有任意数目的与门，并且每一个与门都可以具有任意数目的输入。



(a) 逻辑框图（ANSI标准特殊形状符号）　　(b) 标准矩形轮廓符号

图5.1　与-或逻辑的一个例子。打开文件F05-01检验该操作

一个4输入与-或逻辑电路的真值表如表5.1所示。中间与门的输出(*AB* 列和 *CD* 列)也展示在该表中。

表5.1　图5.1中与-或逻辑的真值表

输入						输出
A	*B*	*C*	*D*	*AB*	*CD*	*X*
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	0	1	1
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	1	1	0	1	1
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	0	1	1
1	1	0	0	1	0	1
1	1	0	1	1	0	1
1	1	1	0	1	0	1
1	1	1	1	1	1	1

与-或电路直接实现乘积项之和表达式，假设变量的反码(如果有)是可用的。图 5.1 中与-或电路的运算如下所述：

对于 4 输入的与-或逻辑电路，如果输入 A 和输入 B 都是高电平(1)或者输入 C 和输入 D 都是高电平(1)，输出 X 就是高电平(1)。

例 5.1　在一个化学品加工厂中，一种液体化学物质被应用在加工过程中。这种化学物质储存在三个不同的储罐中。当任何一个储罐中化学物质的液位降低到某个设定点时，储罐的液位传感器就产生一个高电平电压。

设计一个电路，用以监测每个储罐中的化学物质液位，并指示何时任意两个储罐中的液位降低到设定点以下。

解： 图 5.2 中的与-或电路具有来自储罐 A、B、C 传感器的输入，如图所示。与门 G_1 检测储罐 A 和 B 中的液位，与门 G_2 检测储罐 A 和 C，而与门 G_3 检测储罐 B 和 C。当任意两个储罐中的化学物质液位过低时，某个与门的两个输入就会同时具有高电平电压，从而使得它的输出也是高电平电压，所以最后或门的输出 X 也是高电平电压。这个高电平电压输入随后用于驱动诸如灯泡或者音响报警之类的指示器，如图所示。

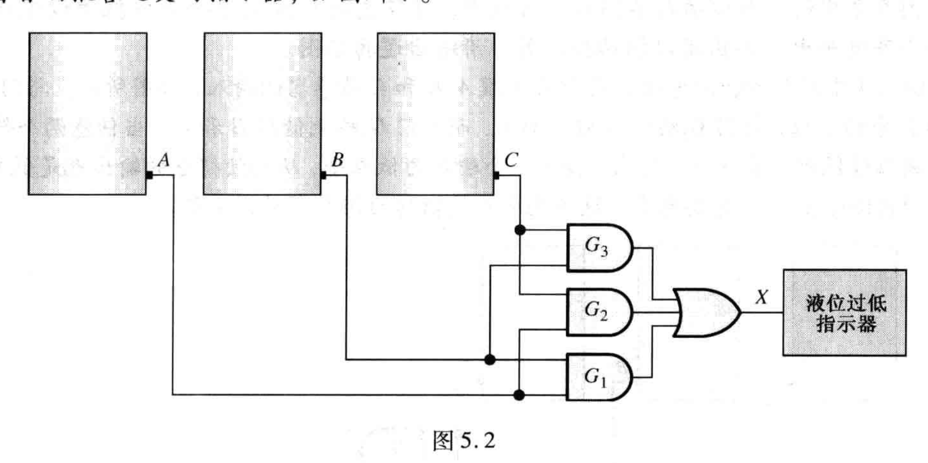


图 5.2

相关问题[①]**：** 写出图 5.2 中与-或逻辑的布尔乘积项之和表达式。

5.1.2　与-或-非逻辑

当与-或电路的输出被求反(反相)后，就会得到一个与-或-非电路。回顾一下，与-或逻辑直接实现乘积项之和表达式。和项之乘积表达式可以用与-或-非逻辑来实现。如下所示，首先给出一个和项之乘积表达式，然后开发出相应的与-或-非表达式。

$$X = (\overline{A} + \overline{B})(\overline{C} + \overline{D}) = (\overline{AB})(\overline{CD}) = \overline{\overline{(\overline{AB})(\overline{CD})}} = \overline{\overline{\overline{AB}}} + \overline{\overline{\overline{CD}}} = \overline{AB + CD}$$

图 5.3(a)中的逻辑框图给出了一个 4 输入的与-或-非电路，并且导出了和项之乘积输出表达式。ANSI 标准矩形轮廓符号如图 5.3(b)所示。一般情况下，一个与-或-非电路可以具有任意数目的与门，且每一个与门可以具有任意数目的输入。

图 5.3 中的与-或-非电路的运算可以做如下表述：

① 答案在本章的结尾。

对于一个 4 输入与-或-非电路，如果输入 A 和输入 B 都是高电平(1)，或者输入 C 和输入 D 都是高电平(1)，那么输出 X 就是低电平。

从表 5.1 的与-或真值表中可以开发出一个与-或-非真值表，只需要将输出列中的所有 1 改为 0、将所有 0 改为 1 即可。

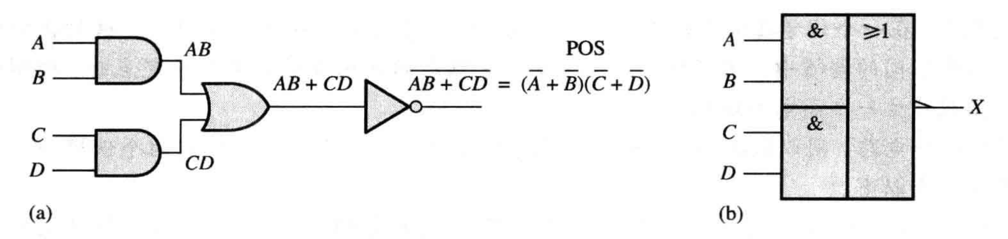


图 5.3 一个产生和项之乘积输出的与-或-非电路。打开文件 F05-03 检验该操作

例 5.2 例 5.1 中的化学物质储罐的传感器被一个新模型所取代，这个模型在储罐中的液位降低到某个设定点时，就会产生一个低电平电压而不是高电平电压。

修改图 5.2 中的电路以运行不同的输入电平，当任意两个储罐中的液位低于设定点时，仍然产生一个高电平电压输出用以驱动指示器。请给出逻辑框图。

解： 图 5.4 中的与-或-非电路具有来自储罐 A、B 和 C 传感器的输入，如图所示。与门 G_1 检测储罐 A 和 B 中的液位，与门 G_2 检测储罐 A 和 C，而与门 G_3 检测储罐 B 和 C。当任意两个储罐中的化学物质液位过低时，每一个与门至少会有一个输入为低电平，从而使得它的输出也是低电平，这样反相器的最终输出 X 就是高电平。这个高电平输出随后用于驱动指示器。

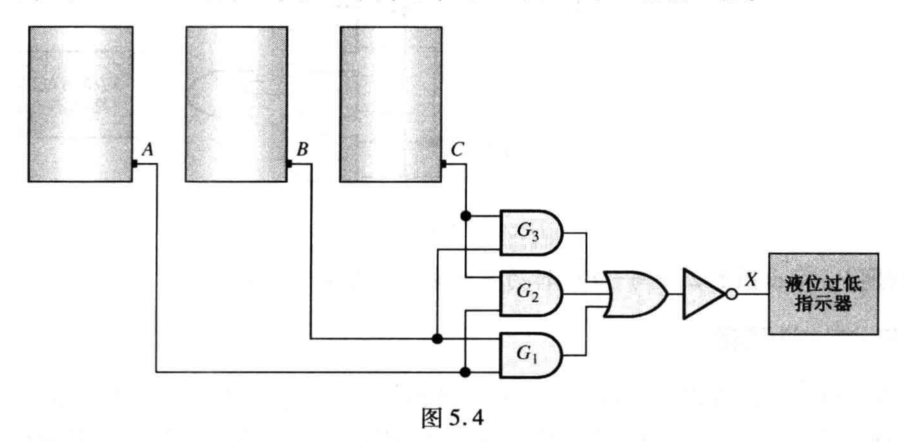


图 5.4

相关问题： 写出图 5.4 中与-或-非逻辑的布尔表达式，并指出当输入 A、B 和 C 中的任意两个都是低电平(0)时，输出就是高电平(1)。

5.1.3 异或逻辑

◇ 异或门实际上就是其他门的一个组合。

在第 3 章介绍了异或门。由于它的重要性，虽然这个电路被当做一种具有自己特殊符号的逻辑门，但是它实际上就是两个与门、一个或门和两个反相器的组合，如图 5.5(a)所示。两个 ANSI 标准逻辑符号如图 5.5(b)和图 5.5(c)所示。

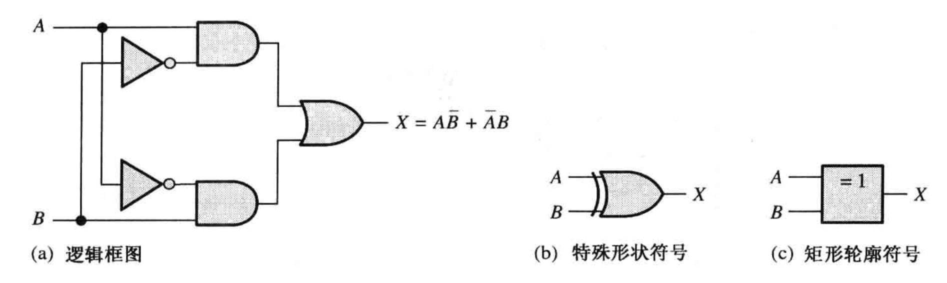


图 5.5　异或逻辑框图和符号。打开文件 F05-05 检验该操作

图 5.5 中电路的输出表达式为

$$X = A\overline{B} + \overline{A}B$$

表 5.2 中的真值表是这个表达式的计算结果。注意只有在两个输入处于相反电平时，输出才是高电平。通常使用一个特殊的异或操作符 $\oplus$，所以表达式 $X = A\overline{B} + \overline{A}B$ 可以表述为"X 等于 A 异或 B"，并且可以写做

$$X = A \oplus B$$

表 5.2　异或逻辑的真值表

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

5.1.4　同或逻辑

异或函数的反码就是同或函数，其推导过程如下所示：

$$X = \overline{A\overline{B} + \overline{A}B} = (\overline{A\overline{B}})(\overline{\overline{A}B}) = (\overline{A} + B)(A + \overline{B}) = \overline{A}\,\overline{B} + AB$$

注意只有当输入 A 和 B 处于相同的电平时，输出 X 才是高电平。

同或逻辑可以通过简单地将异或逻辑输出反相而实现，如图 5.6(a)所示，或者直接由表达式 $\overline{A}\,\overline{B} + AB$ 来实现，如图 5.6(b)所示。

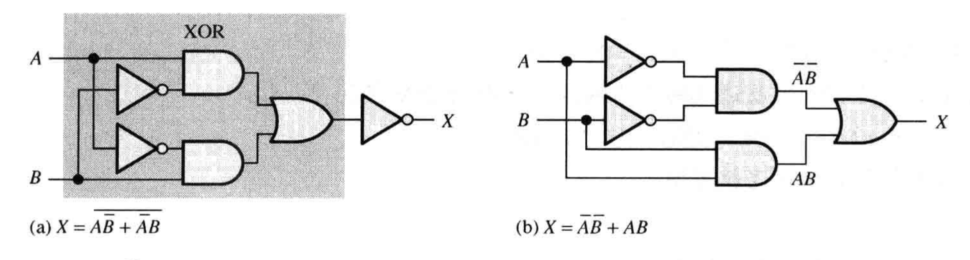


图 5.6　实现同或门的两种等价方法。打开文件 F05-06 检验该操作

例 5.3　使用异或门实现一个 4 位码的偶校验码生成器。

解： 回顾第 2 章，为了给出错误检测，在一个二进制码中加上一个校验位。对于偶校验，在源码中加上一个校验位以使生成的码中 1 的个数为偶数。如图 5.7 的电路所示，当输入中 1 的个数为奇数时，输出为 1，从而使得生成的码中 1 的总数为偶数。当输入中 1 的个数为偶数时，输出为 0。

相关问题： 对于 4 位数据的组合，如何检验产生了一个正确的偶校验位？

例 5.4　使用异或门实现 5 个位码的偶校验生成器，参考例 5.3 的电路。

解： 图 5.8 的电路在 5 位码中出现错误时输出 1，反之输出 0。

相关问题： 当输入码不正确时，如何确认错误？

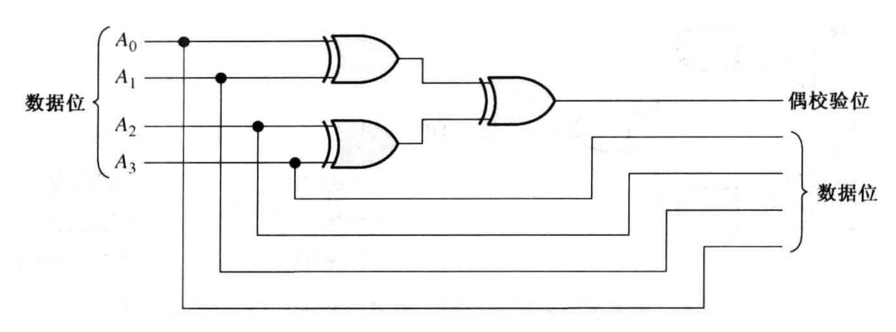


图 5.7　偶校验生成器

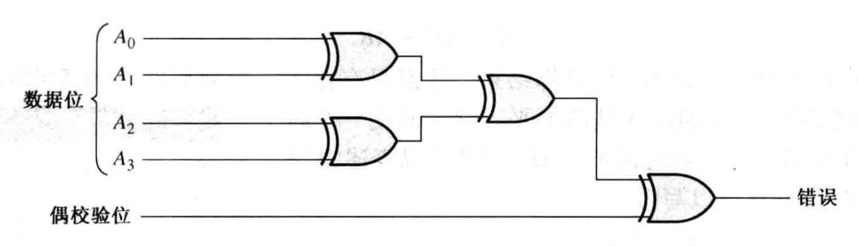


图 5.8　偶校验检测器

5.1 节　温故而知新　(答案在本章的结尾。)

1. 确定如下每个输入的 4 变量与-或-非电路的输出(1 或 0)。
   (a) $A=1$, $B=0$, $C=1$, $D=0$
   (b) $A=1$, $B=1$, $C=0$, $D=1$
   (c) $A=0$, $B=1$, $C=1$, $D=1$
2. 确定如下每个输入的异或门电路的输出(1 或 0)。
   (a) $A=1$, $B=0$　　(b) $A=1$, $B=1$
   (c) $A=0$, $B=1$　　(d) $A=0$, $B=0$
3. 为某个具有输出表达式的 3 输入逻辑电路写出真值表。
   $$X=A\overline{B}C+\overline{A}BC+\overline{A}\overline{B}\overline{C}+AB\overline{C}+ABC$$
4. 为异或电路画出逻辑框图。

5.2　组合逻辑电路的实现

5.2.1　从布尔表达式到逻辑电路

◇ 每一个布尔表达式都有一个相应的逻辑电路，每一个逻辑电路也都有一个相应的布尔表达式。

考虑下面这个布尔表达式：

$$X=AB+CDE$$

简单观察该表达式，就会发现这个表达式由两个项 AB 和 CDE 组成，并且有一个 5 变量的域。第一项由 A 和 B 相与组成，而第二项由 C、D 和 E 相与组成，这两项再进行或运算，从而形成输出 X。这些运算由表达式的结构所指示，如下所示：

与

$$X = AB + CDE$$

或

注意在这个特殊的表达式中，与运算形成的两个独立的项 AB 和 CDE，必须在这两项进行或运算之前执行。

为了实现这个布尔表达式，就需要一个 2 输入与门来形成项 AB，一个 3 输入与门来形成项 CDE。然后还需要一个 2 输入或门来组合这两个项。最后得到的逻辑电路如图 5.9 所示。

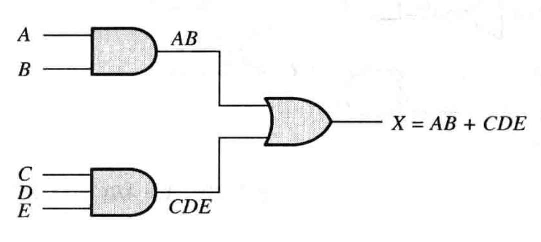


图 5.9　$X = AB + CDE$ 的逻辑电路



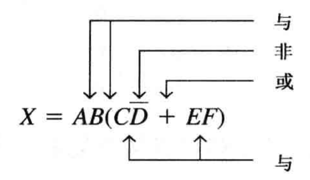
计算机小知识

许多控制程序都需要计算机来执行逻辑运算。驱动程序就是用于计算机外围设备的控制程序。例如，鼠标驱动程序需要进行逻辑检测来确定是否有按钮被按下，并且进一步的逻辑检测将确定其是否被移动(水平方向或者垂直方向)。微处理器的核心部分是算术逻辑单元(ALU)，其执行由程序指令所指定的逻辑运算。对于本章所描述的所有逻辑，如果给出合适的指令，都可以由 ALU 来执行。

实现下面的表达式：

$$X = AB(C\overline{D} + EF)$$

对该表达式的分解展示了 AB 和 $C\overline{D} + EF$ 进行了与运算。$C\overline{D} + EF$ 对 C 和 $\overline{D}$ 及 E 和 F 首先进行与运算，然后再对这两项进行或运算。该结构和表达式的关系如下所示：



在表达式最后形成之前，必须具有生成项 $C\overline{D} + EF$；但是在得到这个项之前，必须具有 $C\overline{D}$ 和 EF；但是在得到 $C\overline{D}$ 之前，必须具有 $\overline{D}$ 。所以，逻辑运算必须以恰当的次序来完成。

实现 $X = AB(C\overline{D} + EF)$ 所需要的逻辑门如下所示：

1. 一个反相器用来形成 $\overline{D}$。
2. 两个 2 输入与门形成 $C\overline{D}$ 和 EF。
3. 一个 2 输入或门形成 $C\overline{D} + EF$。
4. 一个 3 输入与门形成 X。

该表达式的逻辑电路如图 5.10(a)所示。注意在这个电路中(从输入 D 到输出)的输入和输出之间，最多有 4 个门和一个反相器。通过逻辑电路的总的传播延迟时间常常是一个主要考虑因素。传播延迟是可增加的，所以输入和输出之间的门或者反相器越多，传播延迟时间就越大。

除了图 5.8(a)的中间项 $C\overline{D} + EF$ 需要作为其他目的的输出，一般情况下最好将电路化简为它的乘积项之和形式，以便减少总的传播延迟时间。将该表达式转换为如下所示的乘积项之和形式，最后得到的电路如图 5.10(b)所示。

$$AB(C\overline{D} + EF) = ABC\overline{D} + ABEF$$

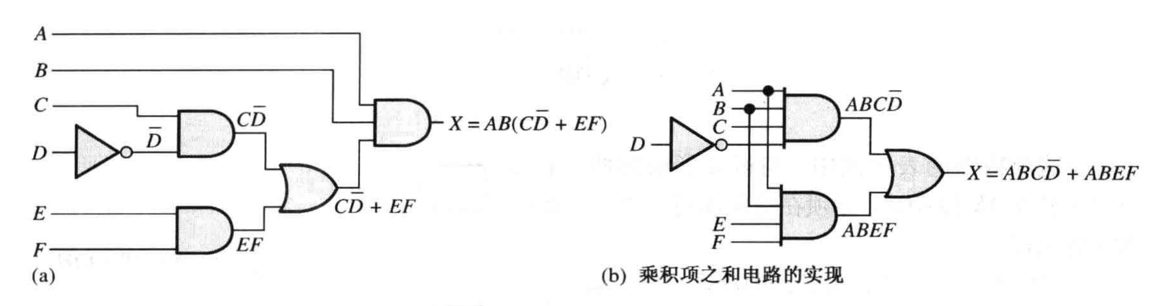


图 5.10 $X = AB(C\overline{D}+EF) = ABC\overline{D}+ABEF$ 的逻辑电路

5.2.2 从真值表到逻辑电路

如果开始于一个真值表而不是表达式，就可以从真值表中写出乘积项之和表达式，然后实现逻辑电路。表 5.3 指定了一个逻辑函数。

表 5.3

输入			输出	
A	B	C	X	乘积项
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	1	$A\overline{B}\overline{C}$
1	0	1	0	
1	1	0	0	
1	1	1	0	

通过把 $X = 1$ 的乘积项进行或运算，就可以从真值表得到乘积项之和表达式：

$$X = \overline{A}BC + A\overline{B}\overline{C}$$

表达式中的第一项通过对 3 个变量 $\overline{A}$、B 和 C 相与而形成。第二项通过对三个变量 A、$\overline{B}$ 和 $\overline{C}$ 相与而形成。

实现该表达式所需要的逻辑门如下所示：3 个反相器以形成变量 $\overline{A}$ 、$\overline{B}$ 和 $\overline{C}$；两个 3 输入与门以形成 $\overline{A}BC$ 和 $A\overline{B}\overline{C}$；以及一个 2 输入或门以形成最终的输出函数 $\overline{A}BC + A\overline{B}\overline{C}$。

该逻辑函数的实现如图 5.11 所示。

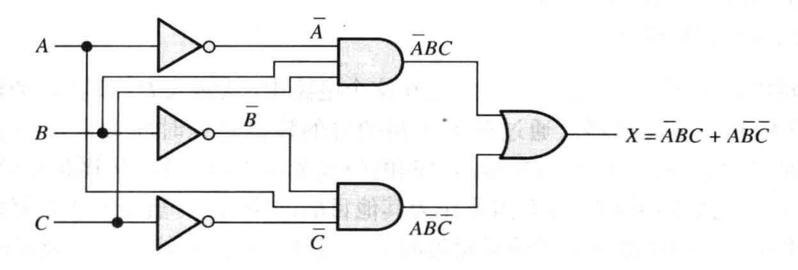


图 5.11 $X = \overline{A}BC + A\overline{B}\overline{C}$ 的逻辑电路。打开文件 F05-11 检验该操作

例5.5 设计一个逻辑电路，完成真值表5.4所确定的运算。

解： 注意仅在3个输入条件下 $X=1$。因此，逻辑表达式是

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C}$$

逻辑门需要3个反相器、3个3输入与门和一个3输入或门。逻辑电路如图5.12所示。

表5.4

输入			输出	
A	B	C	X	乘积项
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	0	
1	0	1	1	$A\overline{B}C$
1	1	0	1	$AB\overline{C}$
1	1	1	0	

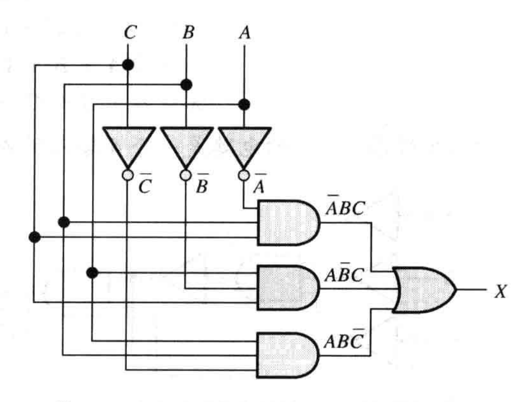


图5.12 打开文件F05-12检验该操作

相关问题： 判断图5.12中的逻辑电路是否能够被化简。

例5.6 设计一个具有4个输入变量的逻辑电路，只有当3个输入变量全为1时，输出才为1。

解： 在4个变量的16种可能的组合中，其中有3个输入为1的组合列在了表5.5中，同时边上列出了相应的乘积项。

乘积项相或以后得到如下的表达式：

$$X = \overline{A}BCD + A\overline{B}CD + AB\overline{C}D + ABC\overline{D}$$

这个表达式由图5.13中的与-或逻辑实现。

表5.5

A	B	C	D	乘积项
0	1	1	1	$\overline{A}BCD$
1	0	1	1	$A\overline{B}CD$
1	1	0	1	$AB\overline{C}D$
1	1	1	0	$ABC\overline{D}$

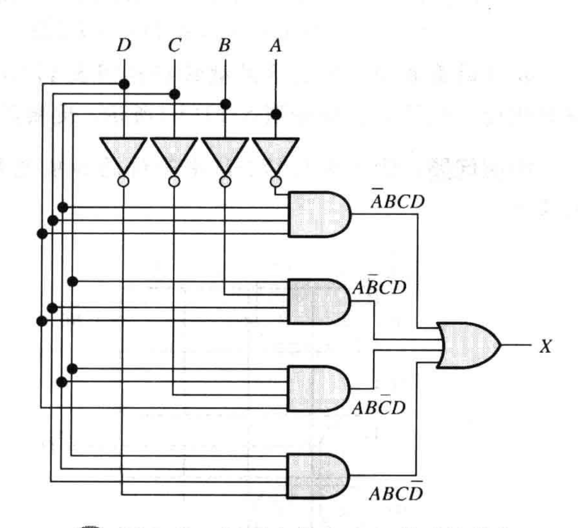


图5.13 打开文件F05-13检验该操作

相关问题： 判断图5.13中的逻辑电路是否能够被化简。

例5.7 把图5.14中的组合逻辑电路化简到最小形式。

解： 输出电路的表达式是

$$X = (\overline{\overline{A}\,\overline{B}\,\overline{C}})C + \overline{\overline{A}\,\overline{B}\,\overline{C}} + D$$

应用狄摩根定理和布尔代数,

$$\begin{aligned}X &= (\overline{\overline{A}} + \overline{\overline{B}} + \overline{\overline{C}})C + \overline{\overline{A}} + \overline{\overline{B}} + \overline{\overline{C}} + D\\ &= AC + BC + CC + A + B + C + D\\ &= AC + BC + C + A + B + \cancel{C} + D\\ &= C(A + B + 1) + A + B + D\\ X &= A + B + C + D\end{aligned}$$

如图 5.15 所示,简化的电路是一个 4 输入的或门。

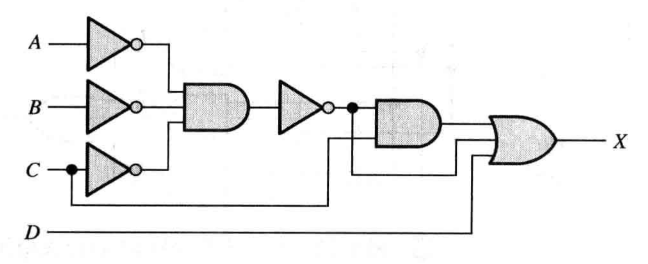


图 5.14 打开文件 F05-14 检验该操作

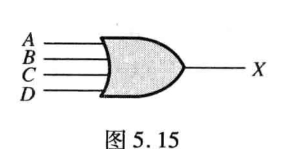


图 5.15

相关问题: 使用卡诺图验证最小表达式 $A + B + C + D$。

例 5.8 最小化图 5.16 中的组合逻辑电路。求反变量的反相器没有画出。

解: 输出表达式是

$$X = A\overline{B}\,\overline{C} + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$$

扩展第一项以包括缺少的变量 D 和 $\overline{D}$,

$$\begin{aligned}X &= A\overline{B}\,\overline{C}(D + \overline{D}) + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\\ &= A\overline{B}\,\overline{C}D + A\overline{B}\,\overline{C}\,\overline{D} + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\end{aligned}$$

这个扩展的乘积项之和表达式被映射到图 5.17(a)的卡诺图上并得到化简。化简的思想如图 5.17(b)所示,反相器没有画出。

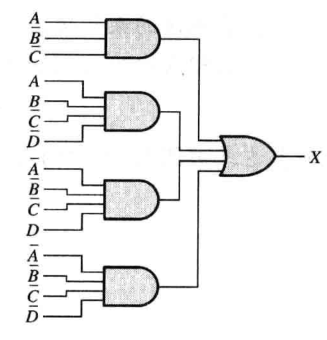


图 5.16

相关问题: 设计图 5.17(b)中等价的和项之乘积电路。参见附录 B。

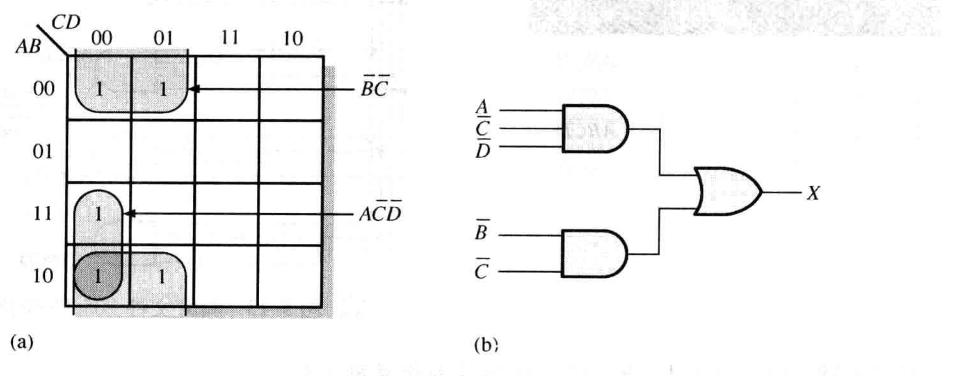


图 5.17

5.2 节　温故而知新

1. 完成如下给出的最小项布尔表达式。
   (a) $X = ABC + AB + AC$　　(b) $X = AB(C + DE)$
2. 设计一个逻辑电路，当所有 3 个输入为 1 或 0 时，输出为 1。
3. 把问题 1 减少到最小乘积项之和的形式，以减少电路元件。

5.3 与非门和或非门的通用特性

5.3.1 与非门作为通用的逻辑元件

◇ 与非门可以用来产生任何逻辑函数。

与非门是一种通用门，因为它可以用来产生反相、与、或和或非的函数。反相器可以通过把与非门的所有输入连接在一起而形成，如图 5.18(a)给出的 2 输入与非门，在效果上相当于一个输入。一个与函数可以仅由与非门产生，如图 5.18(b)所示。一个或函数可以仅由与非门产生，如图 5.18(c)所示。最后，一个或非函数的产生如图 5.18(d)所示。

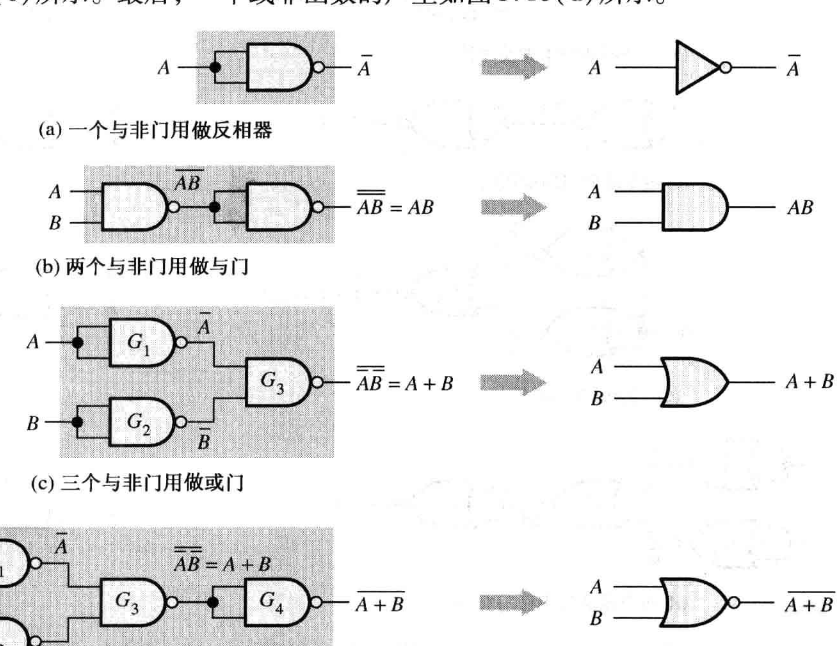


图 5.18　与非门的通用应用。打开文件 F05-18 检验该操作

在图 5.18(b)中，一个与非门用于将前面一个与非门的输出取反(反码)以形成与函数，如下面的等式所示：

$$X = \overline{\overline{AB}} = AB$$

在图 5.18(c)中，门 G_1 和 G_2 用于在两个输入变量进入与非门 G_3 之前取反。最后的或输出应用狄摩根定理推导如下：

$$X = \overline{\overline{A}\,\overline{B}} = A + B$$

在图5.18(d)中，与非门G_4用做反相器，连接到图5.18(c)的电路上以产生或非运算$\overline{A+B}$。

5.3.2 或非门作为通用的逻辑元件

◇ 或非门可以用来产生任何逻辑函数。

如同与非门，或非门可以用来产生反相、与、或和与非的函数。一个非电路或反相器可以通过把或非门的所有输入连接在一起而形成，如图5.19(a)所给出的2输入或非门，在效果上相当于一个输入。同样，一个或门可以由或非门产生，如图5.19(b)所示。一个与门可以使用或非门构造出来，如图5.19(c)所示。这里或非门G_1和G_2用做反相器，最后的输出应用狄摩根定理推导如下：

$$X = \overline{\overline{A} + \overline{B}} = AB$$

图5.19(d)给出了或非门如何形成与非门函数的电路。

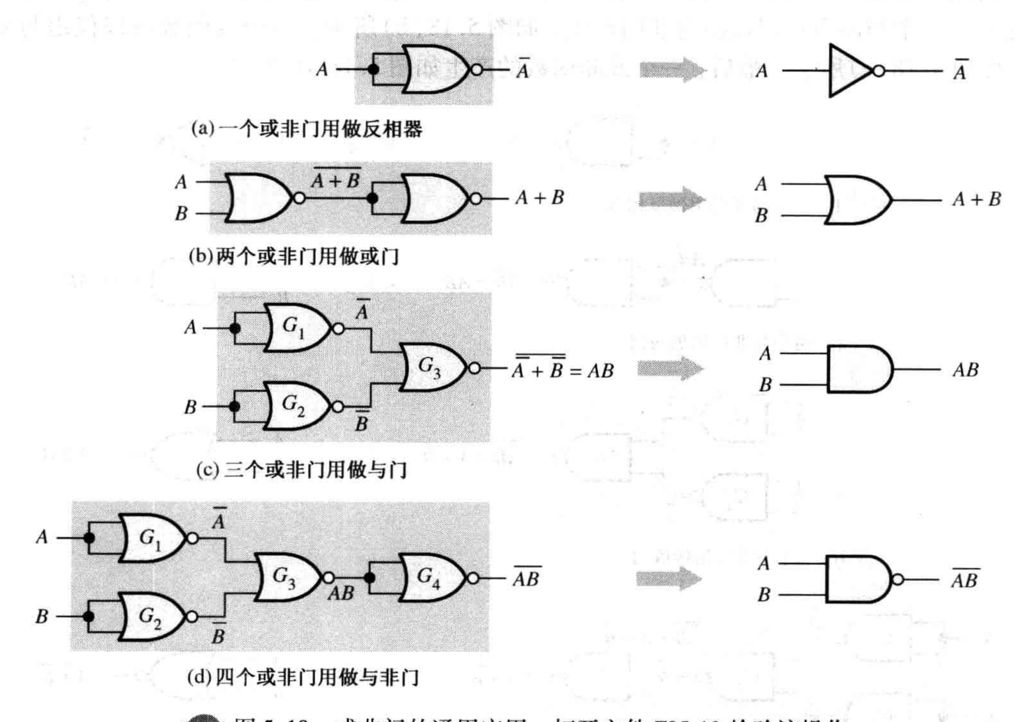


图5.19 或非门的通用应用。打开文件F05-19检验该操作

5.3节 温故而知新

1. 使用与非门实现如下的表达式：

   (a) $X = \overline{A} + B$ (b) $X = A\overline{B}$

2. 使用或非门实现如下的表达式：

   (a) $X = \overline{A} + B$ (b) $X = A\overline{B}$

5.4 使用与非门和或非门的组合逻辑

5.4.1 与非逻辑

正如已经学习的，通过使用狄摩根定理，与非门可以用做与非或者非-或函数。

$$\overline{AB} = \overline{A} + \overline{B}$$

与非 ——↑　　↑—— 非-或

考虑图 5.20 中的与非逻辑。设计输出表达式的步骤如下所示：

$$X = \overline{(\overline{AB})(\overline{CD})}$$
$$= \overline{(\overline{A} + \overline{B})(\overline{C} + \overline{D})}$$
$$= (\overline{\overline{A} + \overline{B}}) + (\overline{\overline{C} + \overline{D}})$$
$$= \overline{\overline{A}}\,\overline{\overline{B}} + \overline{\overline{C}}\,\overline{\overline{D}}$$
$$= AB + CD$$

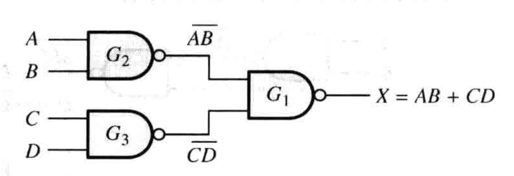
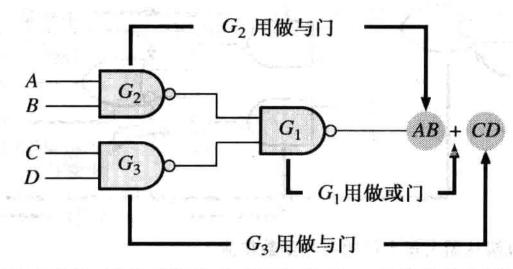


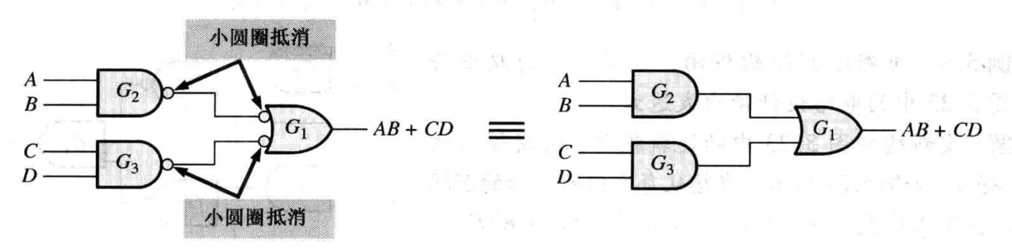
图 5.20　$X = AB + CD$ 的与非逻辑

正如在图 5.20 中所看到的一样，输出表达式 $AB + CD$ 是两个与项被或在了一起。这就展示了门 G_2 和 G_3 用做与门，门 G_1 用做或门，如图 5.21(a)所示。这个电路在图 5.21(b)中重新绘制，对门 G_2 和 G_3 使用了与非符号，而对门 G_1 使用了非-或符号。

注意在图 5.21(b)中门 G_2 和 G_3 的输出与门 G_1 的输入之间小圆圈到小圆圈的连接：由于一个小圆圈表示一个反相，两个连接的小圆圈就表示一个双重反相，所以相互抵消。这种反相抵消可以在前面的输出表达式 $AB + CD$ 的设计中看到，并且由输出表达式中上划线的缺失所指示。因此，图 5.21(b)是有效的与-或电路，如图 5.21(c)所示。



(a) 原始的与非逻辑图给出了和输出表达式有关的有效门运算



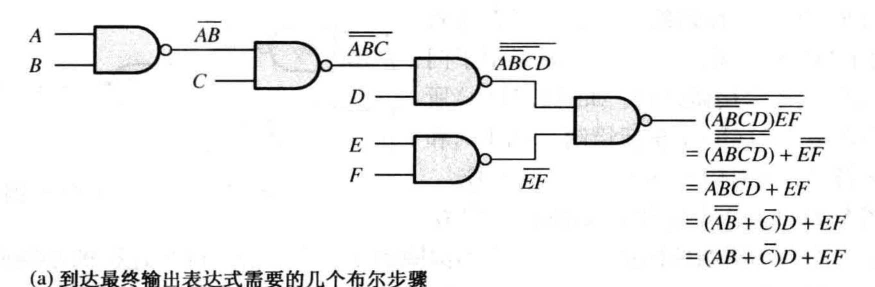
(b) 等价的与非 / 非-或逻辑框图　　(c) 等价的与-或

图 5.21　图 5.20 中电路的与-或等价逻辑的设计

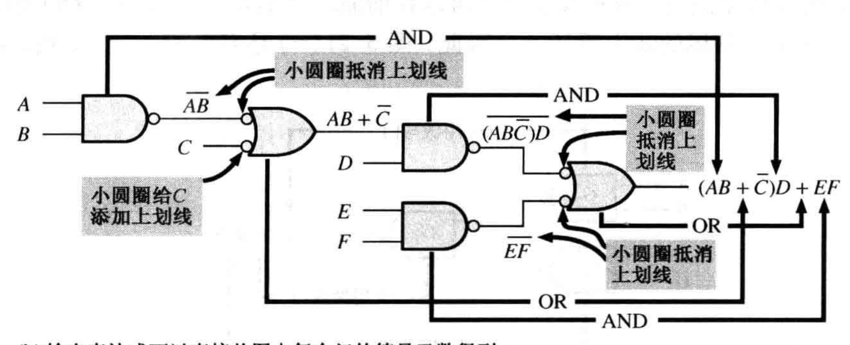
使用双重符号的与非逻辑框图　对于所有使用与非门的逻辑框图，在绘制它们的时候，都要使用一个与非门符号或等价的非-或符号来表示每一个门，以反映逻辑电路中门之间的运算。与非符号和非-或符号称为双重符号。在绘制一个与非逻辑框图时，总是以这样的方法使用门的符号，在一个门的输出和另一个门的输入之间的每个连接要么是小圆圈到小圆圈，要么是非小圆圈到非小圆圈。一般情况下，在逻辑框图中一个小圆圈输出不应连接到非小圆圈输入，反之亦然。

图5.22给出了门的一种排列，用以解释为具有几个门级的与非电路使用适当的双重符号的过程。尽管像图5.22(a)那样使用所有的与非符号是正确的，但图5.22(b)就更加易“读”，因而是更好的方法。如图5.22(b)所示，输出门以非-与符号出现。这时与非符号恰好用在输出门的前面，并且当从输出移开时，后续门的符号是交替的。

门的形状指示了它的输入出现在输出表达式中的方式，因此展示了该门的功能在逻辑电路中的作用如何。对于与非符号来说，输入以与的形式出现在输出表达式中；而对于非-或符号来说，输入以或的形式出现在输出表达式中，如图5.22(b)所示。根据图5.22(b)中的双重符号图，更加容易直接从该逻辑框图中确定输出表达式，因为每个门符号都指示了它的输入变量的关系，因为它们都呈现在输出表达式中。



(a) 到达最终输出表达式需要的几个布尔步骤



(b) 输出表达式可以直接从图上每个门的符号函数得到

图5.22　在一个与非逻辑框图中解释双重符号的恰当使用

例5.9　重新绘制逻辑框图，使用恰当的双重符号为图5.23中的电路设计输出表达式。

解： 重新绘制图5.23中的逻辑框图，使用等价的非-或符号，如图5.24所示。直接从每个门所指示的逻辑运算写出X表达式，得到$X = (\overline{A} + \overline{B})C + (\overline{D} + \overline{E})F$。

相关问题： 从图5.23中导出输出表达式，并展示它等价于解中的表达式。

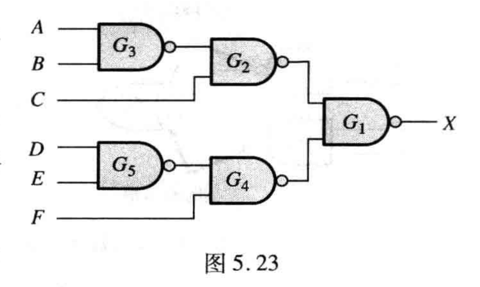


图5.23

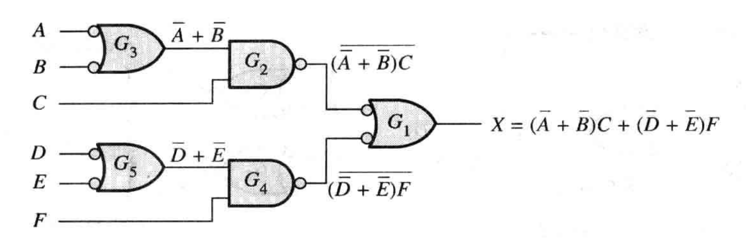


图 5.24

例 5.10　使用恰当的双重符号实现每一个与非逻辑表达式：

(a) $ABC+DE$　　　(b) $ABC+\bar{D}+\bar{E}$

解： 参照图 5.25。

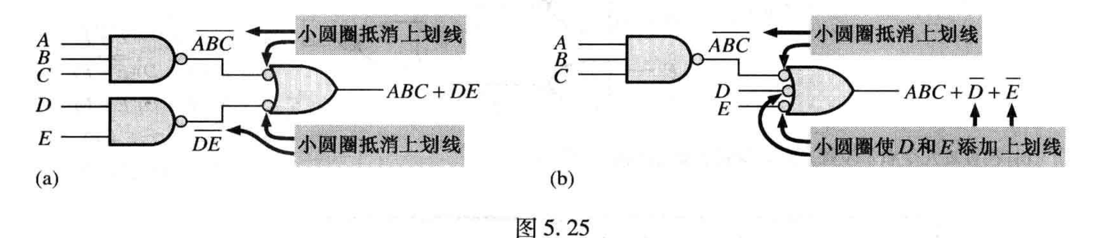


图 5.25

相关问题： 把图 5.25(a)和(b)中的与非电路转换为等价的非-或逻辑。

5.4.2　或非逻辑

或非门可以用做或非或者非-与，如狄摩根定理所展示的那样。

$$\overline{A+B}=\bar{A}\bar{B}$$

或非 ↗　　↖ 非-与

考虑一下图 5.26 中的或非逻辑。输出表达式的设计如下所示：

$$X=\overline{\overline{A+B}+\overline{C+D}}=(\overline{\overline{A+B}})(\overline{\overline{C+D}})=(A+B)(C+D)$$

图 5.26　或非逻辑 $X=(A+B)(C+D)$

如图 5.26 所示，输出表达式$(A+B)(C+D)$由两个或项的相与组成。这就展示了门 G_2 和 G_3 用做或门而门 G_1 用做与门，如图 5.27(a)所示。这个电路在图 5.27(b)中用门 G_1 的非-与符号重新绘制。

使用双重符号的与非逻辑框图　和与非逻辑相同，使用双重符号的目的是使得逻辑框图更加容易理解和分析，如图 5.28 的或非逻辑电路所示。当图 5.28(a)的电路利用双重符号而重新绘制在图 5.28(b)之后，注意门之间的所有输出到输入的连接，要么是小圆圈到小圆圈，要么是非小圆圈到非小圆圈。再次可以看到每个门符号的形状，指示了其在输出表达式中所产生项的类型(与或者或)，因此使得输出表达式更加容易确定，并且使得逻辑框图更加容易分析。

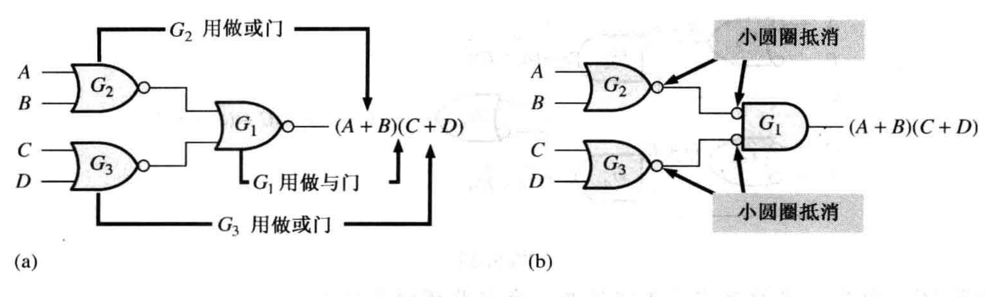
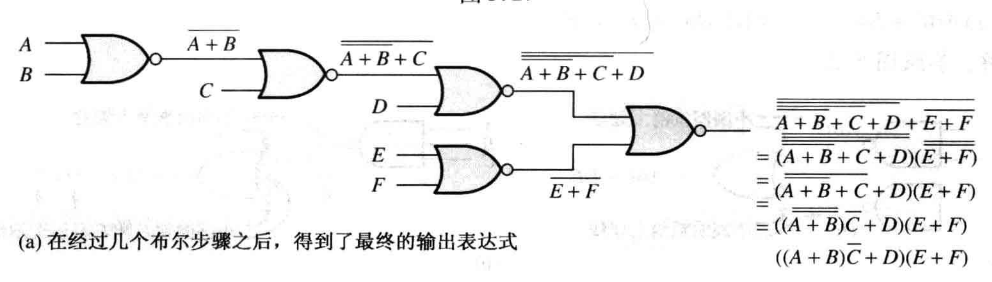


图 5.27



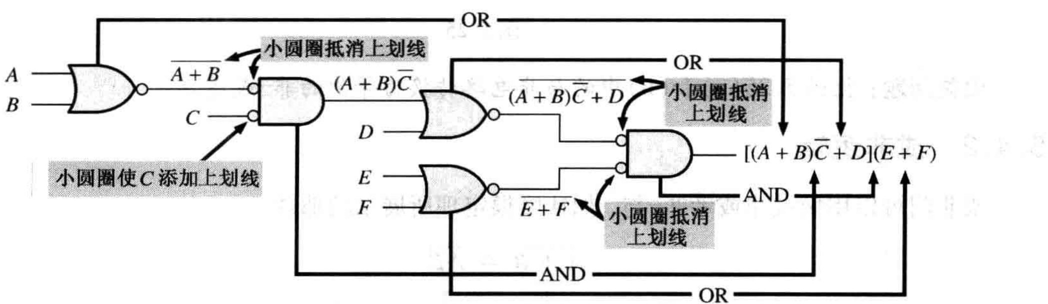


图 5.28　在或非逻辑框图中使用恰当的双重符号

例 5.11　使用恰当的双重符号，重新绘制逻辑框图并设计图 5.29 中电路的输出表达式。

解： 使用等价的非-与符号重新绘制逻辑框图，如图 5.30 所示，直接从每个门所指示的运算中写出 X 的表达式。

$$X = (\overline{A}\,\overline{B} + C)(\overline{D}\,\overline{E} + F)$$

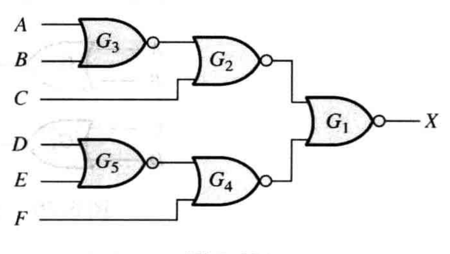


图 5.29

相关问题： 证明图 5.30 中或非电路的输出和图 5.29 中电路的输出是一样的。

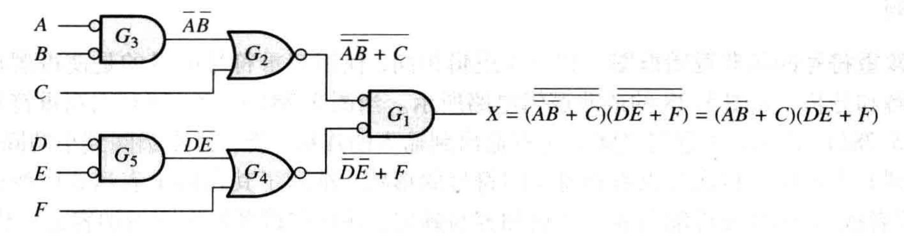


图 5.30

5.4 节　温故而知新

1. 使用与非逻辑设计表达式 $X = \overline{(\overline{A} + \overline{B} + \overline{C})DE}$。
2. 使用或非逻辑设计表达式 $X = \overline{\overline{A}\,\overline{B}\overline{C} + (D + E)}$。

5.5　具有脉冲波形输入的逻辑电路运算

任何门的运算都是相同的，不管它的输入是脉冲还是常量电位。输入的本质(脉冲或者常量电位)并不会改变电路的真值表。本节中的例子将给出具有脉冲输入的组合逻辑电路的分析。

下面是独立门运算的回顾，用以分析具有脉冲波形输入的组合电路：

1. 只有当所有的输入在相同的时刻都是高电平时，与门的输出才是高电平。
2. 只要有一个输入是高电平，或门的输出就是高电平。
3. 只有当所有的输入在相同的时刻都是高电平时，与非门的输出才是低电平。
4. 只要有一个输入是高电平，或非门的输出就是低电平。

例 5.12　确定图 5.31 中电路的最终输出波形 X，输入波形 A、B 和 C 如图所示。

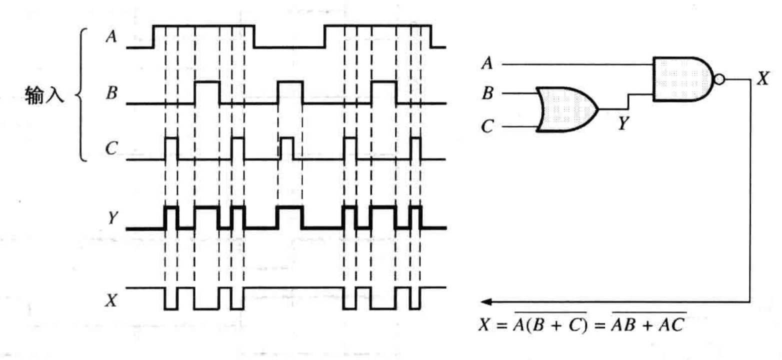


图 5.31

解： 输出表达式 $\overline{AB + AC}$ 指示了当 A 和 B 都是高电平，或者当 A 和 C 都是高电平，或者当所有的输入都是高电平时，输出 X 就是低电平。该输出波形 X 如图 5.31 中的时序图所示。同时还给出了或门输出处中间的波形 Y。

相关问题： 如果输入 A 是一个常量高电平，那么确定其输出波形。

例 5.13　为图 5.32 中的电路绘制时序图，给出 G_1、G_2 和 G_3 的输出，输入波形 A 和 B 如图所示。

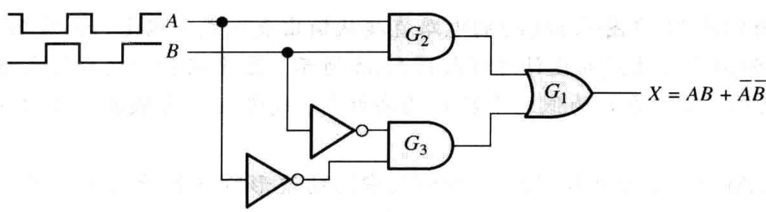


图 5.32

解：当两个输入都是高电平或者当两个输入都是低电平时，输出 X 就是高电平，如图 5.31所示。注意这是一个同或门电路。门 G_2和 G_3的中间输出也在图 5.33 中给出。

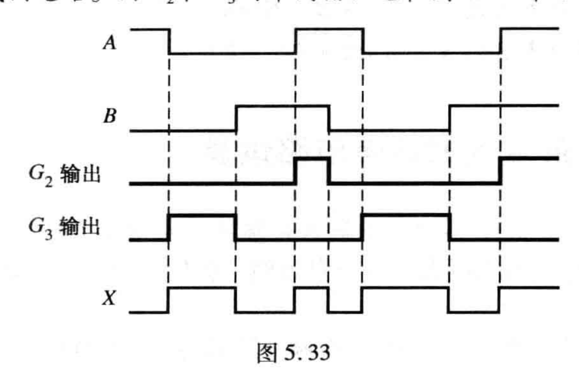


图 5.33

相关问题：如果输入 B 被反相，确定图 5.32 中的输出 X。

例 5.14 为图 5.34(a)所示的逻辑电路确定输出波形 X，首先找到位于每一个点 Y_1、Y_2、Y_3和 Y_4上的中间波形。输入波形如图 5.34(b)所示。

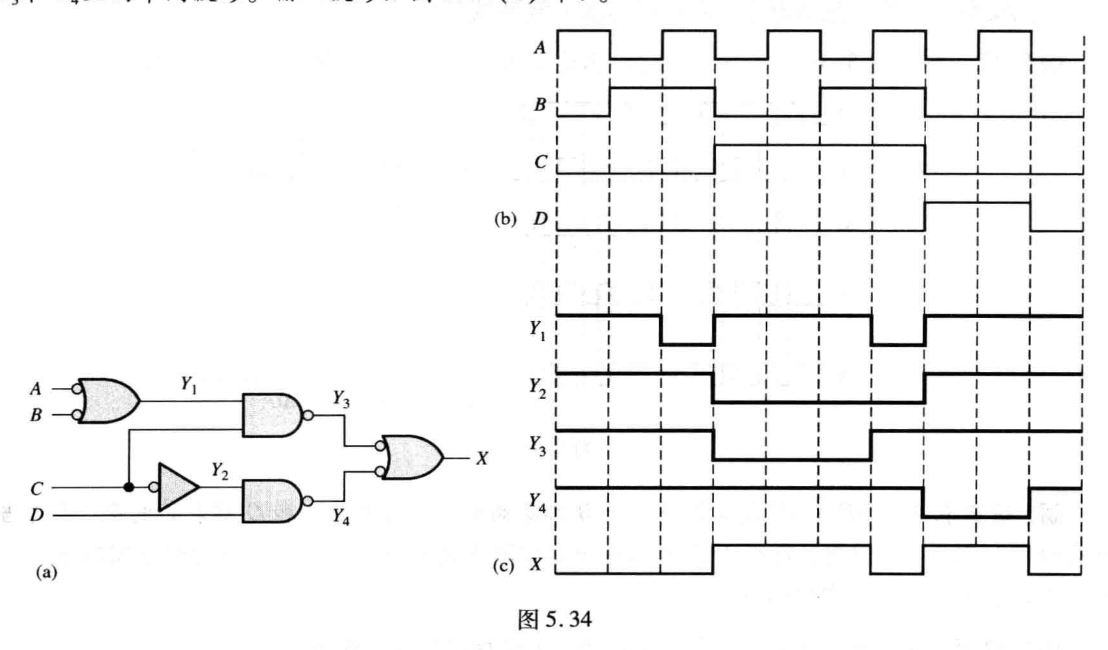


图 5.34

解：所有的中间波形及最终输出波形都在图 5.34(c)中给出。

相关问题：如图输入波形 A 被反相，确定 Y_1、Y_2、Y_3和 Y_4的波形。

例 5.15 为例 5.14 中图 5.34(a)的电路直接从输出表达式中确定输出波形 X。

解：该电路的输出表达式的设计过程如图 5.35 所示。这个乘积项之和形式指示了当 A 为低电平及 C 为高电平，或者为 B 为低电平及 C 为高电平，或者当 C 为低电平及 D 为高电平时，输出就是高电平。

结果如图 5.36 所示，并且和例 5.14 中利用中间的波形所得到的结果一样。每个产生高电平输出的波形条件所对应的乘积项都已展示出来。

相关问题：如果输入波形全部反相，重复此例题。

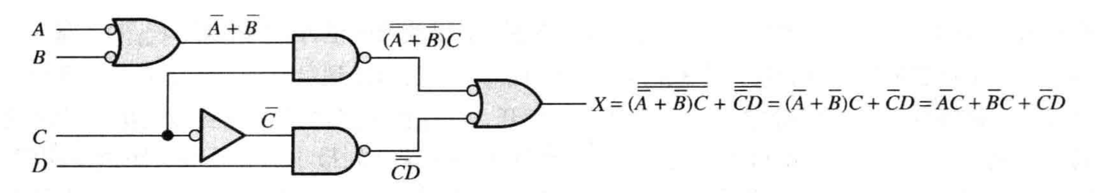


图 5.35

图 5.36

5.5 节　温故而知新

1. 一个宽度 t_W = 50 μs 的脉冲加在异或电路的一个输入上。在第一个脉冲的上升沿开始 15 μs 以后，第二个宽度为 t_W = 10 μs 的脉冲加在另一个输入上。给出对应输入的输出。
2. 图 5.31 中脉冲波形 A 和 B 加在图 5.32 的同或电路上。请画出完整的时序图。



数字系统应用

液罐控制逻辑

图 5.37 是一个食品糖浆生产厂的存储液罐系统。控制逻辑使得灌装的玉米浆预热到一定的温度，以使得在送达混合桶和配料(如糖、调味料、防腐剂和着色剂)加在一起之前，达到适当的黏稠度。液罐里的液位和温度传感器与流量传感器为控制逻辑提供输入。

系统运行和分析

液罐用于存放生产食品糖浆过程中的玉米浆。在混合配料中，当把玉米浆从液罐放入混合桶时，玉米浆必须达到一个具有适当黏稠度的特定温度，以产生所需要的流量特性。这个温度可以通过一个键盘输入来选择。控制逻辑通过打开和关闭加热器来保持温度值。来自温度传感器的模拟输出(T_{analog})由一个模-数转换器把它转换为 8 位二进制码，然后再转换为 8 位 BCD 码。温度控制器检测温度值，当温度低于指定的值后把加热器打开。当温度到达指定的值后把加热器关闭。

当玉米浆的液位到达或高于最大液位时，液位传感器产生一个高电平。阀门控制逻辑检

测最大液位(L_{max})或最小液位(L_{min})两个值是否到达，以及液体是否流向液罐(F_{inlet})。基于这些输入，控制逻辑打开和关闭每个阀门(V_{inlet} 和 V_{outlet})。仅当达到最小液位时，新的玉米浆才可以通过液罐阀门加到液罐中。一旦入口阀门打开，只有在液罐的液位达到最大值后才能关闭入口阀门。同样，一旦出口阀门打开，只有在液罐的液位达到最小值后才能关闭出口阀门。新加入的玉米浆总是比液罐里的要凉一些。在加入玉米浆时或温度低于设定值时，不能从液罐中倒出玉米浆。

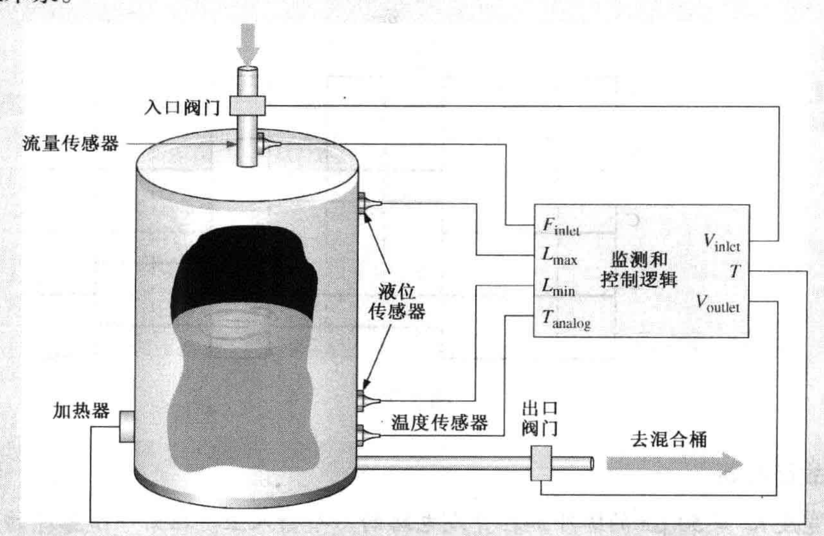


图 5.37 带有液位和温度传感器及控制的液罐

入口阀门控制 入口阀门打开，允许向液罐加入玉米浆的情况如下所示：

- 液位在最小值(L_{min})。
- 液罐在加装(F_{inlet})，并且没有达到最大液位($\overline{L}_{max}$)。

表 5.6 为入口阀门的真值表。入口阀门打开的有效电平为高电平(1)。

表 5.6 入口阀门控制的真值表

输入 L_{max}	L_{min}	F_{inlet}	输出 V_{inlet}	描述
0	0	0	1	液位低于最小值。没有液体流入
0	0	1	1	液位低于最小值。有液体流入
0	1	0	0	液位大于最小值，低于最大值。没有液体流入
0	1	1	1	液位大于最小值，低于最大值。有液体流入
1	0	0	X	不会发生
1	0	1	X	不会发生
1	1	0	0	液位处于最大值。没有液体流入
1	1	1	0	液位处于最大值。有液体流入

1. 为什么在真值表中有两种情况不会发生?
2. 在哪些情况下入口阀门打开?
3. 一旦液位低于最小值和液罐开始灌装后，入口阀门什么时候关闭?

根据真值表，可以写出入口阀门控制输出的表达式：

$$V_{inlet} = \overline{L}_{max}\overline{L}_{min}\overline{F}_{inlet} + \overline{L}_{max}\overline{L}_{min}F_{inlet} + \overline{L}_{max}L_{min}\overline{F}_{inlet}$$

与或表达式可以化简为如下的表达式：

$$V_{inlet} = \overline{L}_{min} + \overline{L}_{max}\overline{F}_{inlet}$$

4. 使用卡诺图，证明简化的表达式是正确的。

5. 使用简化的表达式，为入口阀门控制画出逻辑框图。

出口阀门控制　出口阀门打开，允许液罐流出玉米浆的情况如下所示：

- 玉米浆的液位大于最小设定点并且液罐没有装满。
- 玉米浆的温度达到设定值。

表 5.7 为出口阀门值的真值表。出口阀门打开的有效电平为高电平(1)。

表 5.7　出口阀门控制的真值表

输入				输出	
L_{max}	L_{min}	F_{inlet}	T	V_{outlet}	描述
0	0	0	0	0	液位低于最小值。没有液体流入，温度低
0	0	0	1	0	液位低于最小值。没有液体流入，温度正确
0	0	1	0	0	液位低于最小值。有液体流入，温度低
0	0	1	1	0	液位低于最小值。有液体流入，温度正确
0	1	0	0	0	液位大于最小值，低于最大值。没有液体流入，温度低
0	1	0	1	1	液位大于最小值，低于最大值。没有液体流入，温度正确
0	1	1	0	0	液位大于最小值，低于最大值。有液体流入，温度低
0	1	1	1	0	液位大于最小值，低于最大值。有液体流入，温度正确
1	0	0	0	X	不会发生
1	0	0	1	X	不会发生
1	0	1	0	X	不会发生
1	0	1	1	X	不会发生
1	1	0	0	0	液位处于最大值。没有液体流入，温度低
1	1	0	1	1	液位处于最大值。没有液体流入，温度正确
1	1	1	0	0	液位处于最大值。有液体流入，温度低
1	1	1	1	0	液位处于最大值。有液体流入，温度正确

6. 为什么出口阀门控制需要 4 个输入而入口阀门仅需要 3 个？

7. 在哪些输入情况下出口阀门开启？

8. 一旦液位到达最大值并且液罐的液体开始流出，什么时间出口阀门关闭？

根据真值表，出口阀门控制的表达式可以写为

$$V_{outlet} = \overline{L}_{max}L_{min}\overline{F}_{inlet}T + L_{max}L_{min}\overline{F}_{inlet}T$$

与或表达式可以化简为如下的表达式：

$$V_{outlet} = L_{min}\overline{F}_{inlet}T$$

9. 使用卡诺图，证明简化的表达式是正确的。

10. 使用简化的表达式，为出口阀门控制画出逻辑框图。

阀门控制逻辑的计算机仿真

图 5.38 为入口和出口阀门控制逻辑的计算机仿真。SPDT 开关用来表示液位、流量传感器的输入和温度指示。探针用来指示输出状态。

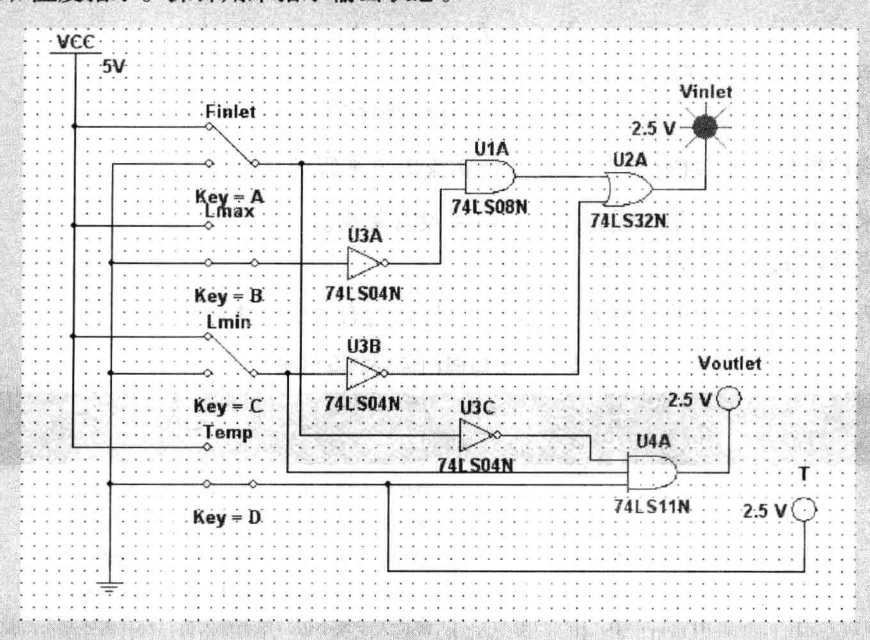


图 5.38 阀门控制逻辑的计算机仿真电路

温度控制 温度控制逻辑接收 8 位 BCD 码，表示测量的温度，并和指定温度的 BCD 码进行比较。图 5.39 给出控制逻辑的框图。模-数转换器和二进制-BCD 码转换器的分析不在应用举例的范围内。

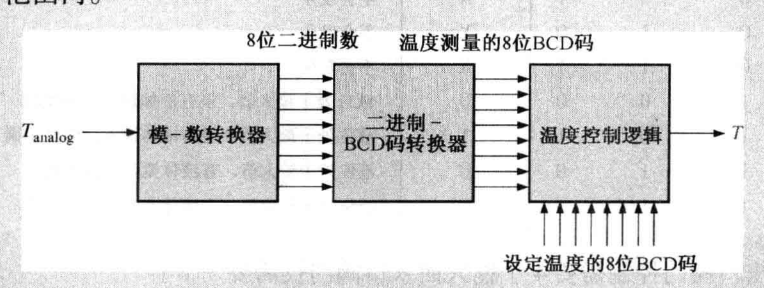


图 5.39 温度控制电路的框图

当测量温度和设定温度相同时，两个 BCD 码相同，并且输出 T 为低电平(0)。当测量上温度低于设定温度时，两个 BCD 码不同，输出 T 为高电平(1)，打开加热器。温度控制逻辑可以由异或门实现，如图 5.40 所示。两个 BCD 码的每一对相应的位加在一个异或门的输入上。如果这两位相同，则异或门的输出为 0；如果不同，则输出为 1。当一个或多个异或门的输出为 1 时，或门的输出 T 等于 1，从而打开加热器。

WWW 打开网络资源系统应用示例的文件 SAA05。使用 Multisim 软件运行阀门控制逻辑仿真程序，观察运行结果。建立一个新的 Multisim 文件，连接温度控制逻辑，并且运行仿真程序。

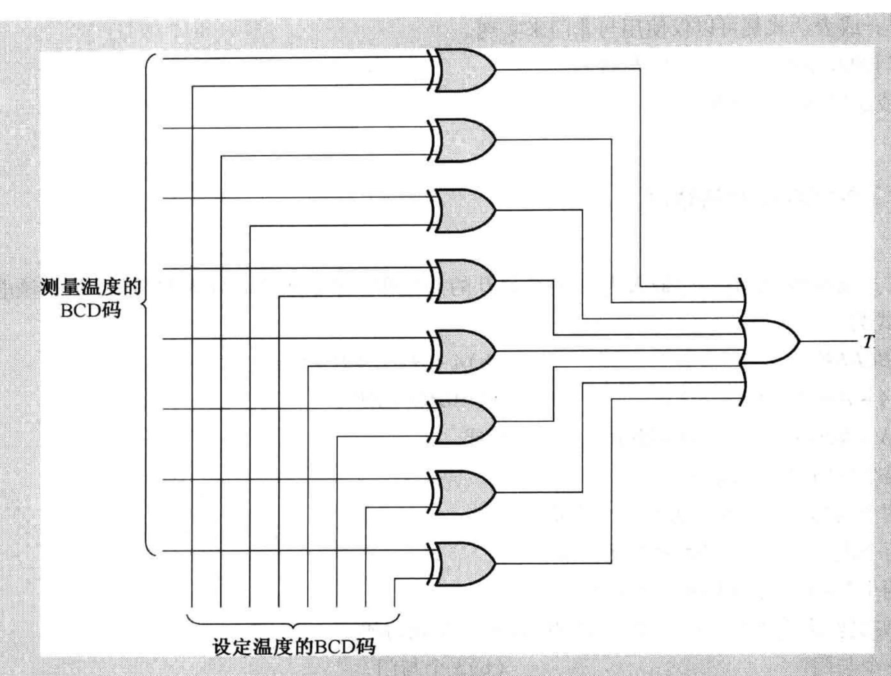


图5.40 温度控制逻辑的框图

把知识用于实践

如果玉米浆的温度不能低于设定温度9℃，那么温度控制逻辑可以简化吗？如果可以，应该怎么实现？

关键词

非-与 当输入为低电平有效时，一个或门的双重操作。

非-或 当输入为低电平有效时，一个与非门的双重操作。

通用门 与非门或者或非门。通用指的是一个门的特性，这个特性允许使用这个门或这些门的组合来实现任何逻辑功能。

判断题 （答案在本章的结尾。）

1. 与-或逻辑可以只有两个2输入与门。
2. AOI是与-或-非的首字母组合词。
3. 如果一个异或门的输入相同，那么输出为高电平(1)。
4. 如果一个同或门的输入不相同，那么输出为低电平(0)。
5. 一个奇偶校验生成器可以使用异或门来实现。
6. 与非门不能用来产生与的功能。
7. 或非门不能用来产生与的功能。

8. 任何与-或表达式都可以仅使用与非门来实现。
9. 与非门的双重符号是一个非-与符号。
10. 非-或逻辑等价于与非门。

自测题 (答案在本章的结尾。)

1. 一个与-或电路，具有一个输入为A、B、C和D的与门和一个输入为E和F的与门，那么该电路的输出表达式为
   (a) $ABCDEF$　　(b) $A+B+C+D+E+F$
   (c) $(A+B+C+D)(E+F)$　　(d) $ABCD+EF$
2. 输出$X=A\overline{B}C+A\overline{C}$的逻辑电路由________组成。
   (a)两个与门和一个或门
   (b)两个与门、一个或门和两个反相器
   (c)两个或门、一个与门和两个反相器
   (d)两个与门、一个或门和一个反相器
3. 为了实现表达式$\overline{A}BCD+A\overline{B}CD+AB\overline{C}\overline{D}$，需要一个或门和________。
   (a)一个与门　　(b)3个与门
   (c)3个与门和4个反相器　　(d)3个与门和3个反相器
4. 表达式$\overline{A}BCD+ABC\overline{D}+A\overline{B}\overline{C}D$
   (a)不能化简　　(b)可以化简为$\overline{A}BC+A\overline{B}$
   (c)可以化简为$ABC\overline{D}+\overline{A}B\overline{C}$　　(d)这些答案都不对
5. 某个与-或非电路具有一个输入为A、B、C和D的与门，以及一个输入为E和F的与门，那么它的输出表达式为
   (a) $ABCD+EF$　　(b) $\overline{A}+\overline{B}+\overline{C}+\overline{D}+\overline{E}+\overline{F}$
   (c) $\overline{(A+B+C+D)(E+F)}$　　(d) $(\overline{A}+\overline{B}+\overline{C}+\overline{D})(\overline{E}+\overline{F})$
6. 异或函数表示为
   (a) $\overline{A}\,\overline{B}+AB$　　(b) $\overline{A}B+A\overline{B}$
   (c) $(\overline{A}+B)(A+\overline{B})$　　(d) $(\overline{A}+\overline{B})+(A+B)$
7. 与运算可以由下面哪一项来产生?
   (a)两个与非门　　(b)3个与非门
   (c)一个或非门　　(d)3个或非门
8. 或运算可以由下面哪一项来产生?
   (a)两个或非门　　(b)3个与非门
   (c)4个与非门　　(d)答案(a)和(b)
9. 当在逻辑框图中使用双重符号时，________。
   (a)小圆圈输出连接小圆圈输入　　(b)与非门符号产生与运算
   (c)非-或符号产生或运算　　(d)所有这些答案都是对的
   (e)没有一个答案是正确的
10. 所有的布尔表达式都可以由什么来实现?
   (a)只用与非门　　(b)只用或非门
   (c)与非门和或非门的组合　　(d)与门、或门和反相器的组合
   (e)上述的任何一种

习题（奇数题的答案在本书的结尾。）

5.1 节　基本组合逻辑电路

1. 为 3-wide(每个设备具有三个与门)4 输入的与-或-非电路绘制 ANSI 特殊形状逻辑框图。同时绘制 ANSI 标准的矩形轮廓符号。

2. 为图 5.41 中的每一个电路写出输出表达式。

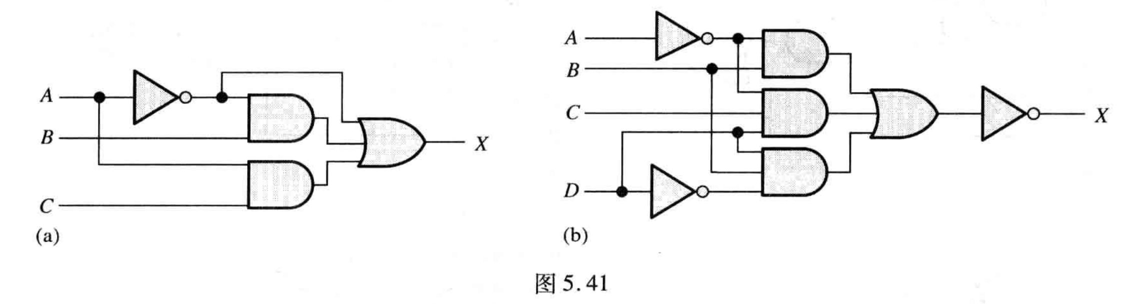


图 5.41

3. 为图 5.42 中的每一个电路写出输出表达式。

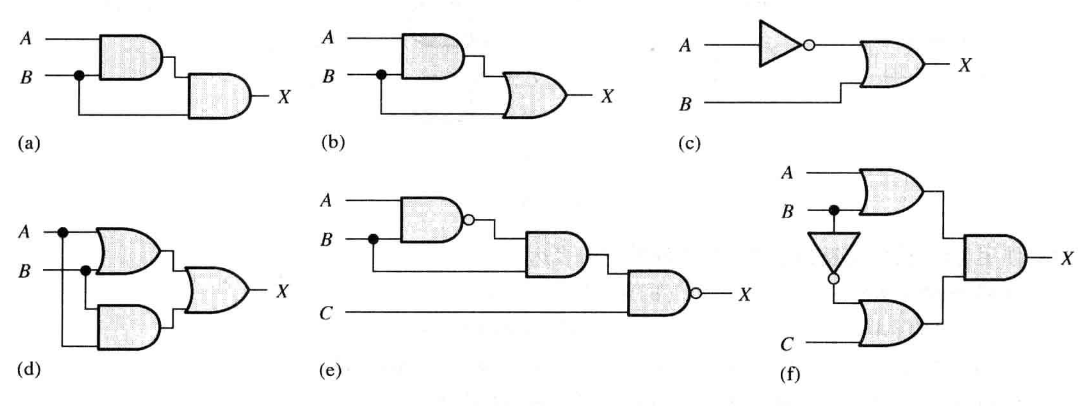


图 5.42

4. 为图 5.43 中的每一个电路写出输出表达式，然后将每个电路改为等价的与或形式。

5. 为图 5.42 中的每一个电路开发真值表。

6. 为图 5.43 中的每一个电路开发真值表。

7. 给出一个产生和项之乘积输出的同或门电路。

5.2 节　组合逻辑电路的实现

8. 为电锯设计一个与-或-非逻辑电路，如果防护装置不在位(逻辑 0)或电机太热(逻辑 1)时开关打开(逻辑 1)。

9. 一个 AOI(与-或-非)逻辑芯片有两个 4 输入与门连接到一个 2 输入或门。为此电路(假设输入为 A 到 H)写出布尔表达式。

10. 使用与门、或门或者两者的组合实现下面的逻辑表达式：

(a) $X = AB$　　(b) $X = A + B$

(c) $X = AB + C$　　(d) $X = ABC + D$

(e) $X = A + B + C$　　(f) $X = ABCD$

(g) $X = A(CD + B)$　　(h) $X = AB(C + DEF) + CE(A + B + F)$

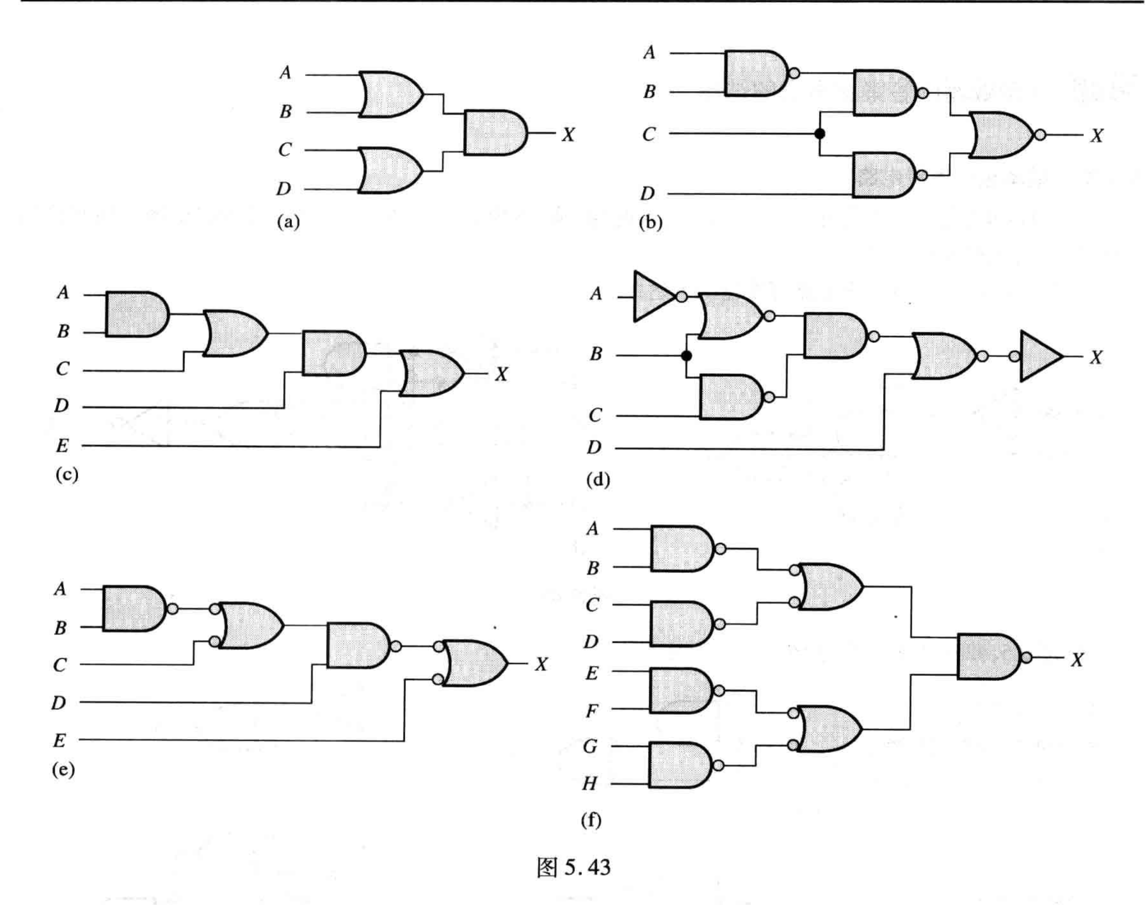


图 5.43

11. 使用与门、或门及反相器来实现下面的逻辑表达式：
    (a) $X = AB + \overline{B}C$　　(b) $X = A(B + \overline{C})$
    (c) $X = A\overline{B} + AB$　　(d) $X = \overline{ABC} + B(EF + \overline{G})$
    (e) $X = A[BC(A + B + C + D)]$　　(f) $X = B(C\overline{D}E + \overline{E}FG)(\overline{AB} + C)$
12. 使用与非门、或非门或者两者的组合来实现下面的逻辑表达式：
    (a) $X = \overline{A}B + CD + (\overline{A + B})(ACD + \overline{BE})$
    (b) $X = AB\overline{CD} + D\overline{E}F + \overline{AF}$
    (c) $X = \overline{A}[B + \overline{C}(D + E)]$
13. 为表 5.8 中的真值表实现逻辑电路图。
14. 为表 5.9 中的真值表实现逻辑电路图。
15. 尽可能地化简图 5.44 中的电路，并且通过给出相同的两个真值表，验证简化的电路和原来的电路等价。
16. 为图 5.45 中的电路重复习题 15。
17. 使用最少的门完成习题 11 乘积项之和形式的每道题的函数。
18. 使用最少的门完成习题 12 乘积项之和形式的每道题的函数。
19. 使用最少的门完成图 5.43 乘积项之和形式的每个电路的函数。

5.3 节　与非门和或非门的通用特性

20. 只用与非门实现图 5.41 中的逻辑电路。
21. 只用与非门实现图 5.45 中的逻辑电路。
22. 只使用或非门重复习题 20。
23. 只使用或非门重复习题 21。

表 5.8

输入			输出
A	B	C	X
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

表 5.9

输入				输出
A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

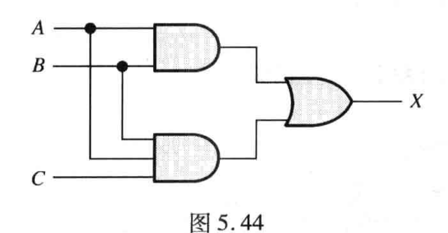


图 5.44

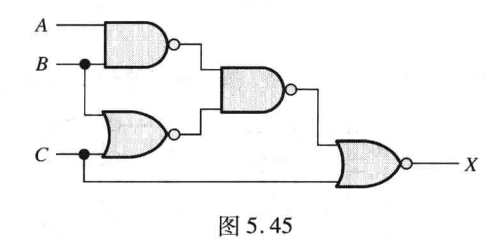


图 5.45

5.4 节　使用与非门和或非门的组合逻辑

24. 给出只使用或非门实现如下表达式的方法：

(a) $X = ABC$　　(b) $X = \overline{ABC}$

(c) $X = A + B$　　(d) $X = A + B + \overline{C}$

(e) $X = \overline{AB} + \overline{CD}$　　(f) $X = (A + B)(C + D)$

(g) $X = AB[C(\overline{DE} + \overline{AB}) + \overline{BCE}]$

25. 只使用与非门重复习题 24。
26. 只使用与非门实现习题 10 中的每一个函数。
27. 只使用与非门实现习题 11 中的每一个函数。

5.5 节　具有脉冲波形输入的逻辑电路运算

28. 给定如图 5.46 所示的逻辑电路和输入波形，绘制输出波形。

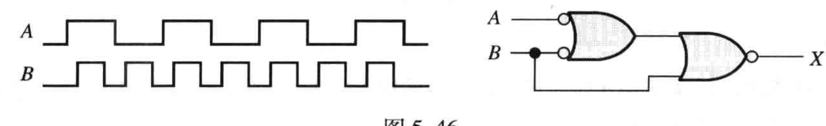


图 5.46

29. 对于图 5.47 中的逻辑电路，绘制出正确的与输入相关的输出波形。
30. 对于图 5.48 中的输入波形，什么样的逻辑电路将会产生所给出的输出波形？

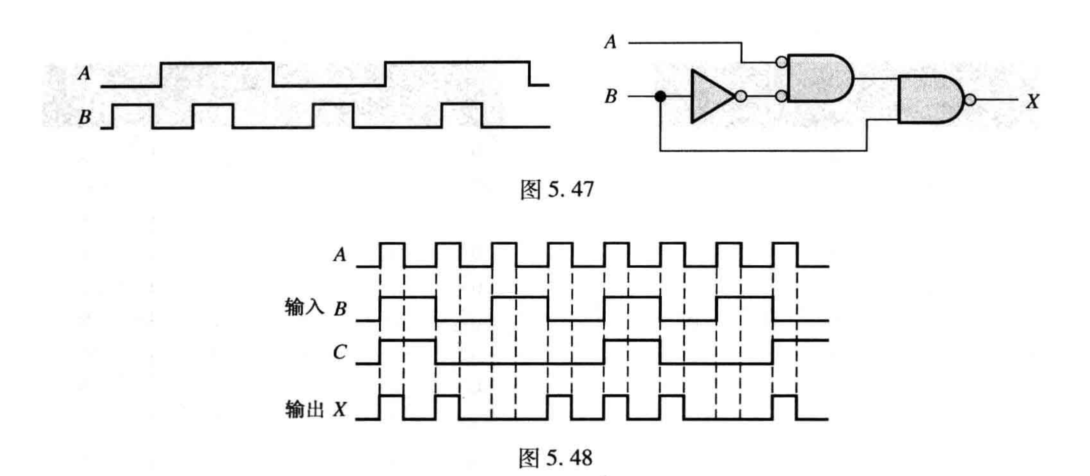


图 5.47

图 5.48

31. 为图 5.49 中的波形重复习题 30。

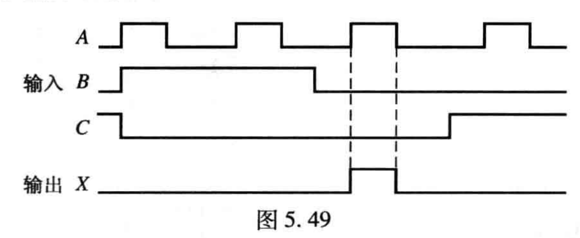


图 5.49

32. 对于图 5.50 中的电路，按照适当的相互关系绘制标有数字的点上的波形。

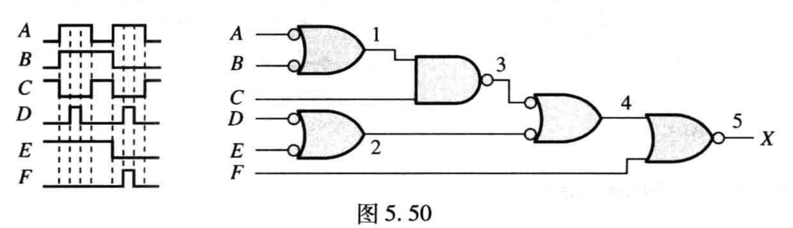


图 5.50

33. 假设穿过每个门的传播延迟是 10 纳秒(ns)，确定图 5.51 中(具有最小宽度 $t_W = 25$ ns 的脉冲如图所示)所需要的输出波形 X 是否可以由给出的输入产生。

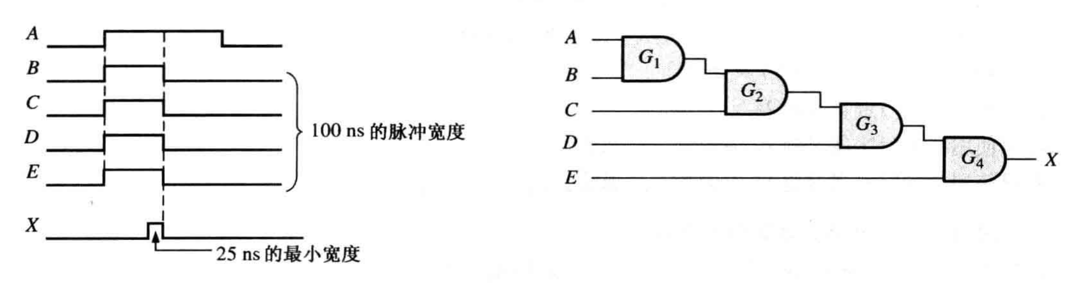


图 5.51

特殊设计问题

34. 设计一个逻辑电路，如果由 4 位二进制数表示的输入大于 12 或小于 3，输出就是高电平。首先列出真值表，然后画出逻辑框图。

35. 开发一个必须满足如下要求的逻辑电路，一个房间里有一盏使用电池的灯，它由两个开关控制，一个在前门，另一个在后门。当前门开关打开并且后门开关关闭，或者前门开关关闭并且后门开关打

开时，电灯打开。当两个开关都关闭或都打开时，电灯关闭。设高电平输出表示打开的情况，低电平输出表示关闭的情况。

36. 使用与非门开发一个十六进制的键盘译码器，把每个按下的键转换成二进制数。

答案

温故而知新

5.1 节　基本组合逻辑电路

1. (a) $\overline{AB+CD}=\overline{1\cdot 0+1\cdot 0}=1$　(b) $\overline{AB+CD}=\overline{1\cdot 1+0\cdot 1}=0$
   (c) $\overline{AB+CD}=\overline{0\cdot 1+1\cdot 1}=0$
2. (a) $A\overline{B}+\overline{A}B=1\cdot\overline{0}+\overline{1}\cdot 0=1$　(b) $A\overline{B}+\overline{A}B=1\cdot\overline{1}+\overline{1}\cdot 1=0$
   (c) $A\overline{B}+\overline{A}B=0\cdot\overline{1}+\overline{0}\cdot 1=1$　(d) $A\overline{B}+\overline{A}B=0\cdot\overline{0}+\overline{0}\cdot 0=0$
3. 当 ABC = 000、011、11、110 和 111 时 X = 1；当 ABC = 001、010 和 100 时 X = 0 。
4. $X = AB + \overline{A}\overline{B}$；参见图 5.6(b)的框图，电路由两个与门、一个或门和两个反相器组成。

5.2 节　组合逻辑电路的实现

1. (a) $X = ABC + AB + AC$：3 个与门，一个或门。
   (b) $X = AB(C + DE)$：3 个与门，一个或门。
2. $X = ABC + \overline{A}\,\overline{B}\overline{C}$：两个与门，一个或门，3 个反相器。
3. (a) $X=AB(C+1)+AC=AB+AC$　(b) $X=AB(C+DE)=ABC+ABDE$

5.3 节　与非门和或非门的通用特性

1. (a) $X = \overline{A} + B$：一个 2 输入与非门，输入为 A 和 $\overline{B}$ 。
   (b) $X = A\overline{B}$：一个 2 输入与非门，输入为 A 和 $\overline{B}$，紧接着使用一个与非门作为反相器。
2. (a) $X = \overline{A} + B$：一个 2 输入或非门，输入为 $\overline{A}$ 和 B，紧接着使用一个或非门作为反相器。
   (b) $X = A\overline{B}$：一个 2 输入或非门，输入为 $\overline{A}$ 和 B。

5.4 节　使用与非门和或非门的组合逻辑

1. $X = \overline{(\overline{A} + \overline{B} + \overline{C})DE}$：一个 3 输入与非门，输入为 A、B 和 C，输出连接到第二个 3 输入与非门，并且有两个其他的输入 D 和 E 。
2. $X = \overline{\overline{A}\,\overline{B}\overline{C} + (D + E)}$：一个 3 输入或非门，输入为 A、B 和 C，输出连接到第二个 3 输入或非门，并且有两个其他的输入端 D 和 E 。

5.5 节　具有脉冲波形输入的逻辑电路运算

1. 异或门的输入为一个 15 μs 的脉冲，紧接着一个 25 μs 的脉冲，两个脉冲之间的间隔为 10 μs。
2. 当一个同或门的两个输入都为高电平或低电平时，它的输出为高电平。

例题的相关问题

5.1　$X=AB+AC+BC$

5.2　$X=\overline{AB+AC+BC}$

$A=0$ 和 $B=0$，$X=\overline{0\cdot 0+0\cdot 1+0\cdot 1}=\overline{0}=1$

$A=0$ 和 $C=0$，$X=\overline{0\cdot 1+0\cdot 0+1\cdot 0}=\overline{0}=1$

$B=0$ 和 $C=0$，$X=\overline{1\cdot 0+1\cdot 0+0\cdot 0}=\overline{0}=1$

5.3 确定所有16个组合的奇偶校验位。每个组合包含的校验位应该有偶数个1。

5.4 使用奇数个1的码，并且检验输出为1。

5.5 不能化简。

5.6 不能化简。

5.7 $X = A + B + C + D$ 有效。

5.8 参见图5.52。

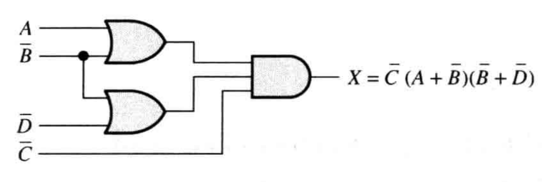


图5.52

5.9 $X = \overline{(\overline{\overline{AB}C})(\overline{\overline{DE}F})} = (\overline{AB})C + (\overline{DE})F = (\overline{A} + \overline{B})C + (\overline{D} + \overline{E})F$

5.10 参见图5.53。

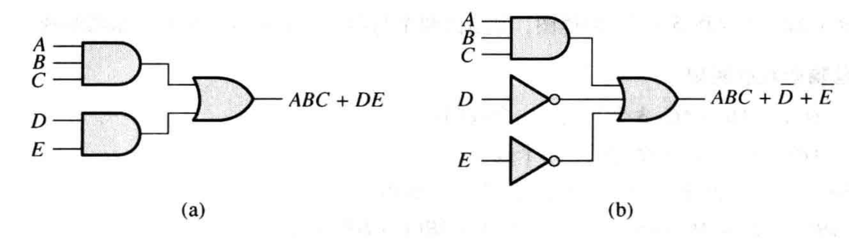


图5.53

5.11 $X = \overline{(\overline{\overline{A+B}+C}) + (\overline{\overline{D+E}+F})} = (\overline{A+B}+C)(\overline{D+E}+F) = (\overline{A}\overline{B}+C)(\overline{D}\overline{E}+F)$

5.12 参见图5.54。

5.13 参见图5.55。

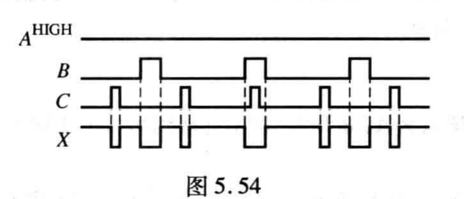


图5.54

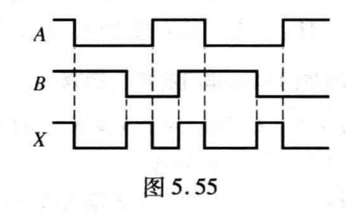


图5.55

5.14 参见图5.56。

5.15 参见图5.57。

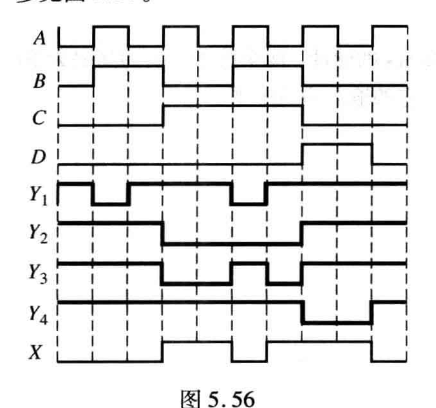


图5.56

图5.57

判断题

1. F　2. T　3. F　4. T　5. T　6. F　7. T　8. T　9. F　10. T

自测题

1. (d)　2. (b)　3. (c)　4. (a)　5. (d)
6. (b)　7. (a)　8. (d)　9. (d)　10. (e)

第6章　组合逻辑电路函数

章节提纲

- 6.1　基本加法器
- 6.2　并行二进制加法器
- 6.3　异步进位与超前进位加法器
- 6.4　比较器
- 6.5　译码器
- 6.6　编码器
- 6.7　代码转换器
- 6.8　多路复用器(数据选择器)
- 6.9　多路分配器
- 6.10　奇偶发生器/校验器
- 数字系统应用

6.1　基本加法器

6.1.1　半加器

◇ 一个半加器进行两位相加，产生一个和及一个进位输出。

回顾第2章所讲到的二进制相加的基本规则。

$$
\begin{aligned}
0+0&=\ \ 0\\
0+1&=\ \ 1\\
1+0&=\ \ 1\\
1+1&=10
\end{aligned}
$$

实现半加运算的逻辑电路称为半加器(half-adder)。

一个半加器的输入为两个二进制数，并在输出产生两个二进制数：一个和位和一个输出进位。

如图6.1所示是用逻辑符号表示的一个半加器。

半加器逻辑　根据表6.1所给出的半加器的运算，和及进位作为输入函数的表达式可以通过推导获得。注意只有当 A 和 B 都为1时，输出进位 C_{out} 才为1；所以 C_{out} 可表示为输入变量的与。

$$C_{out}=AB \tag{6.1}$$

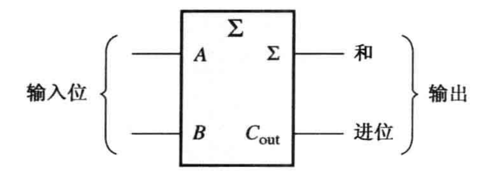


图6.1　半加器的逻辑符号。打开文件F06-01检验该操作

表6.1　半加器真值表

A	B	C_{out}	Σ
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Σ = 和
C_{out} = 输出进位
A 和 B = 输入变量(操作数)

注意，只有当输入变量 A 与 B 不相等时，和输出(Σ)才为1。所以和可以用输入变量的异或来表示。

$$\Sigma = A \oplus B \tag{6.2}$$

根据式(6.1)和式(6.2)，可以设计出满足半加器功能的逻辑电路。A 和 B 输入的与门产生进位输出，异或门产生一个和输出，如图6.2所示。记住异或门是由与门、或门和反相器实现的。

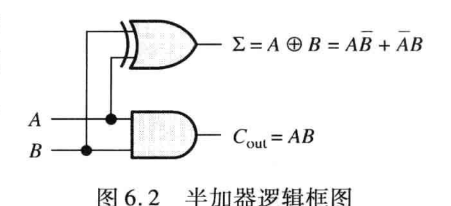


图6.2　半加器逻辑框图

6.1.2　全加器

◇ 全加器有一个输入进位，而半加器没有。

另一类加法器就是全加器。

全加器有两个输入位和一个输入进位，它产生一个和输出及一个输出进位。

全加器和半加器的主要区别就是全加器有一个输入进位。全加器的逻辑符号如图6.3所示，全加器运算的真值表如表6.2所示。

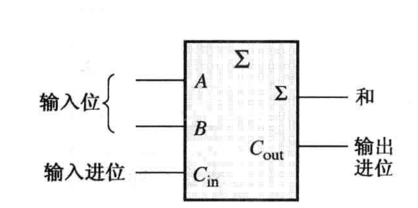


图6.3　全加器的逻辑符号。打开文件F06-03检验该操作

表6.2　全加器真值表

A	B	C_{in}	C_{out}	Σ
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

C_{in} = 输入进位，有时定义为 CI
C_{out} = 输出进位，有时定义为 CO
Σ = 和
A 和 B = 输入变量(操作数)

全加器逻辑　全加器必须将两个输入位和一个输入进位相加。从半加器中可以知道，输入位 A 和 B 的和就是这两个变量的异或 $A \oplus B$ 。由于进位输入(C_{in})是与两个输入位的和相加，那么它一定是与 $A \oplus B$ 异或，这样就得到全加器输出和的等式。

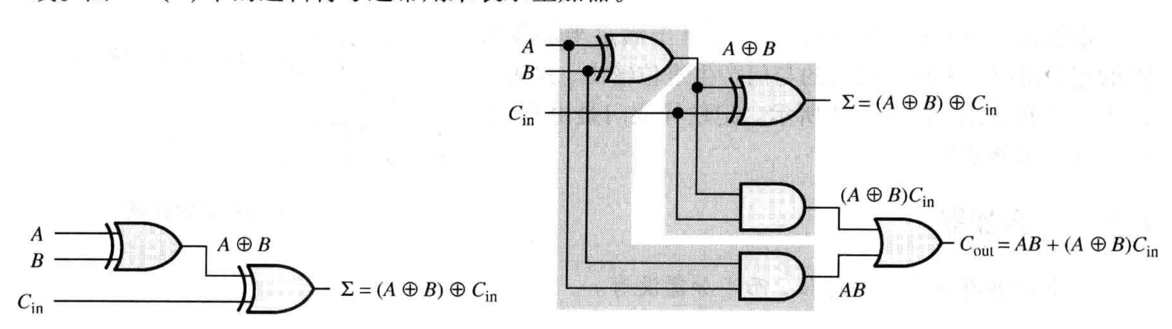
$$\boldsymbol{\Sigma = (A \oplus B) \oplus C_{in}} \tag{6.3}$$

这就意味着要实现全加器的和函数，需要使用两个2输入异或门。第一个产生项 $A \oplus B$，第二个用第一个异或门的输出和输入进位作为输入，如图6.4(a)所示。

当第一个异或门的所有输入都为1时，或当第二个异或门的所有输入都为1时，进位为1。可以根据表6.2来验证这个结论。输入 A 和 B 相与再加上 $A \oplus B$ 和 C_{in} 相与，产生了全加器的输出，如式(6.4)所示的两个与项相或。这个函数加上和逻辑组成了一个完整的全加器电路，如图6.4(b)所示。

$$C_{out} = AB + (A \oplus B)C_{in} \tag{6.4}$$

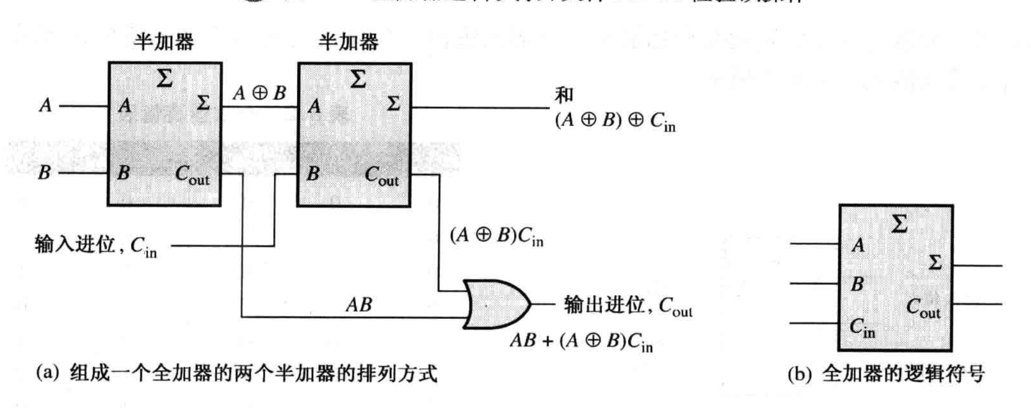
请注意，图6.4(b)中有两个半加器，它们连接图6.5(a)所示的模块，即两个输出进位相或。图6.5(b)中的逻辑符号通常用来表示全加器。



(a) 形成三个位的和所需的逻辑

(b) 实现全加器的完整逻辑电路（每一个阴影区域是一个半加器）

图6.4 全加器逻辑。打开文件F06-04检验该操作



(a) 组成一个全加器的两个半加器的排列方式

(b) 全加器的逻辑符号

图6.5 利用半加器组成全加器

例6.1 如图6.6所示，这3个全加器中的每一个全加器的输入已给出，请确定它们的输出。

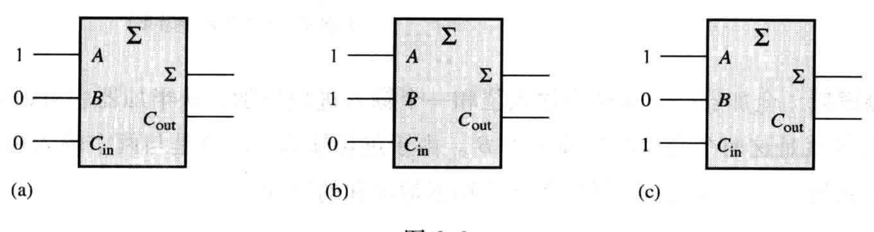


图6.6

解: (a)位输入为 $A = 1$、$B = 0$ 和 $C_{in} = 0$。

$$1 + 0 + 0 = 1 \quad \text{没有进位}$$

所以，$\Sigma = \mathbf{1}$ 和 $C_{out} = \mathbf{0}$。

(b)位输入为 $A = 1$、$B = 1$ 和 $C_{in} = 0$。

$$1 + 1 + 0 = 0 \quad \text{有进位 } 1$$

所以，$\Sigma = \mathbf{0}$ 和 $C_{out} = \mathbf{1}$。

(c)位输入为 $A = 1$、$B = 0$ 和 $C_{in} = 1$。

$$1+0+1=0 \quad \text{有进位 } 1$$

所以，$\Sigma=\mathbf{0}$ 和 $C_{out}=\mathbf{1}$。

相关问题①：当 $A=1$、$B=1$、$C_{in}=1$ 时，全加器的输出是什么？

6.1 节　温故而知新（答案在本章的结尾。）

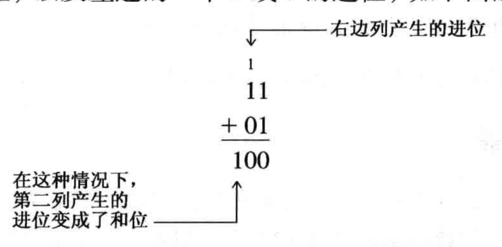
1. 确定如下每个输入位的半加器的和（Σ）和进位（C_{out}）。

(a)01　(b)00　(c)10　(d)11

2. 一个全加器有 $C_{in}=1$，当 $A=1$ 和 $B=1$ 时，它的和输出（Σ）和进位（C_{out}）输出是多少？

6.2 并行二进制加法器

在 6.1 节已经知道，一个单一的全加器能够把两个 1 位数字和一个进位输入相加。为了把多于一位的二进制数相加，就必须使用更多的全加器。当一个二进制数与另一个二进制数相加时，每一列会产生一个和位，以及左边的一个 1 或 0 的进位，如下面的两位数字所示。





计算机小知识

计算机进行一次加法运算的两个数，称为操作数。保存在 ALU（算术逻辑单元）寄存器（如累加器）中已存在的数称为目的操作数，和它相加的操作数称为源操作数。这两个数相加以后又被保存在累加器中。完成整数或小数的加法运算，可以分别使用 ADD 或者 FADD 指令。

如果要将两个二进制数相加，数字中的每一个位都需要一个全加器。那么 2 位数字需要两个加法器；4 位数字需要 4 个加法器；以此类推。每一个加法器的进位输出与下一个较高位的加法器的进位输入相连，如图 6.7 所示的 2 位加法器。注意最低位有效位可以使用一个半加器，或者使用一个全加器，它的进位输入为 0（接地），因为最低有效位是没有进位输入的。

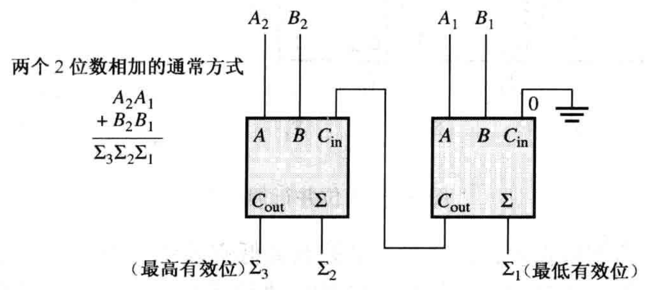


图 6.7　使用两个全加器组成的一个基本 2 位并行加法器框图。打开文件 F06-07 检验该操作

① 相关问题的答案在本章的结尾。

在图6.7中，用A_1和B_1来表示这两个数字的最低有效位(LSB)。下一个较高的位用A_2和B_2来表示。3个和位用Σ_1、Σ_2和Σ_3来表示。注意，全加器最左边的进位输出变成了加法结果中的最高有效位(MSB)，即Σ_3。

例6.2 在图6.8中，求3位并行加法器产生的和，并给出当二进制数101和011相加时中间的进位。

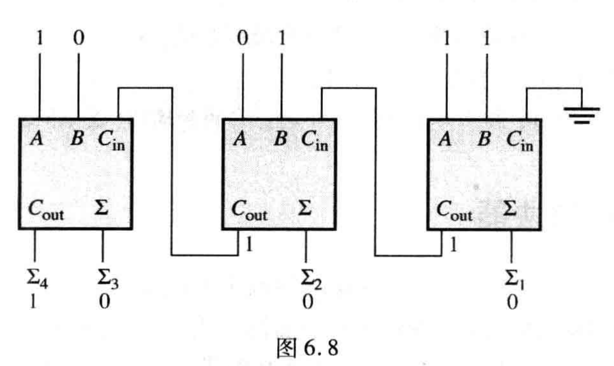


图6.8

解： 这两个数的最低有效位是全加器最右边的位。和位与中间进位如图6.8所示。

相关问题： 当111和101用3位并行加法器相加时，其和输出是什么？

6.2.1 4位并行加法器

4位一组称为一个半字节(nibble)。在图6.9中，4个全加器组成了一个基本4位并行加法器。如前所述，相加的每一个数中的最低有效位(A_1和B_1)在最右边的全加器上相加，较高一级的位在接续的较高一级的全加器上相加，相加的每一个数中最高有效位(A_4和B_4)在最左边的全加器上相加。如图所示，每一个加法器的进位输出与下一个较高一级的加法器的进位输入相连。这些进位称为内部进位。

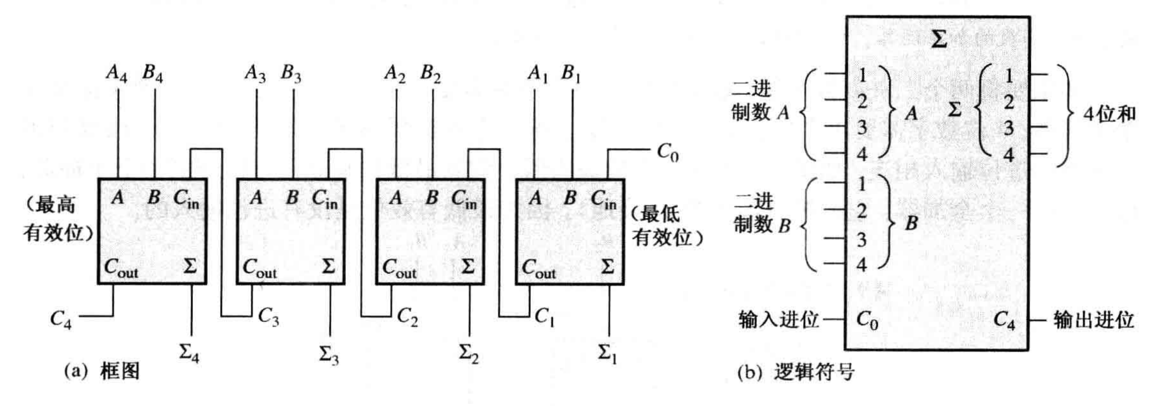


图6.9 4位并行加法器

为了与大多数厂商的数据表保持一致，把处在最低有效位的加法器的进位输入记为C_0；在有4位的情况下，处在最高有效位的加法器的输出进位就是C_4；最低有效位到最高有效位就是相加后的输出。逻辑符号如图6.9(b)所示。

在一个并行加法器中，处理进位的方法有两种：异步进位加法器和超前进位加法器。这将在6.3节讨论。

6.2.2　4位并行加法器的真值表

4位加法器的真值表如表6.3所示。在有些数据表中，真值表称为函数表或函数真值表。下标n表示加法器的位，它可以是4位加法器的1、2、3或4位，C_{n-1}是来自前一级加法器的进位。C_1、C_2和C_3通常是内部进位。C_0是外部进位输入，C_4为输出。例6.3解释表6.3的使用。

表6.3　4位并行加法器的每一级真值表

C_{n-1}	A_n	B_n	Σ_n	C_n
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

例6.3　使用4位并行加法器的真值表(见表6.3)，求出下列两个4位数字的和及进位输出，如果进位输入C_{n-1}是0：

$$A_4A_3A_2A_1 = 1100 \quad 和 \quad B_4B_3B_2B_1 = 1100$$

解：对于$n=1$：$A_1=0$、$B_1=0$和$C_{n-1}=0$。从真值表的第1行可得

$$\Sigma_1 = \mathbf{0} \quad 和 \quad C_1 = 0$$

对于$n=2$：$A_2=0$、$B_2=0$和$C_{n-1}=0$。从真值表的第1行可得

$$\Sigma_2 = \mathbf{0} \quad 和 \quad C_2 = 0$$

对于$n=3$：$A_3=1$、$B_3=1$和$C_{n-1}=0$。从真值表的第4行可得

$$\Sigma_3 = \mathbf{0} \quad 和 \quad C_3 = 1$$

对于$n=4$：$A_4=1$、$B_4=1$和$C_{n-1}=1$。从真值表的最后一行可得

$$\Sigma_4 = \mathbf{1} \quad 和 \quad C_4 = \mathbf{1}$$

C_4成为进位输出，1100与1100的和为11000。

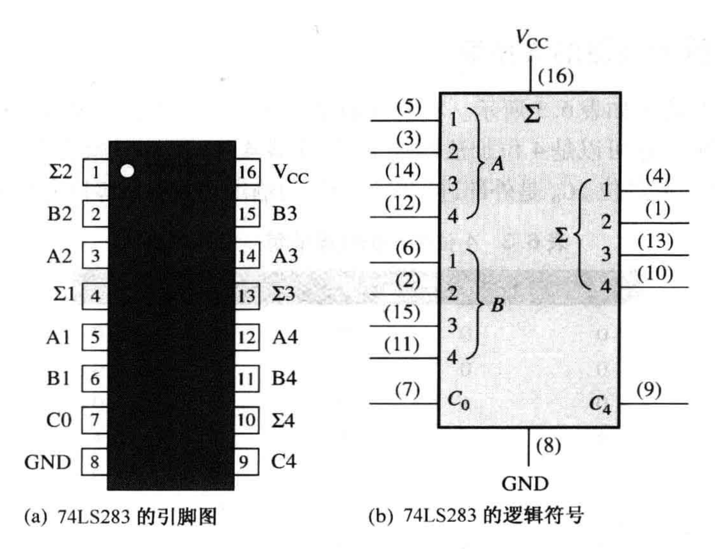
相关问题：使用真值表(见表6.3)求二进制数1011和1010的和。



4位并行加法器74LS283

4位并行加法器实例的集成电路芯片有74LS283。74LS283的标准封装结构是，引脚16是V_{CC}，引脚8接地。图6.10为芯片的引脚图和逻辑符号，逻辑符号的括号里是引脚编号。

集成电路数据表的特点　回顾前述，逻辑门有一个从输入到输出的特定的传输延迟时间t_P。对于集成电路逻辑来说，可能有几个不同规格的t_P。如图6.11所示，4位并行加法器有4种规格的t_P，这只是74LS283数据表的一部分。



(a) 74LS283 的引脚图　　(b) 74LS283 的逻辑符号

图 6.10　4 位并行加法器

符号	参数	限制			单位
		最小	类型	最大	
t_{PLH} t_{PHL}	传输延迟，C_0输入到任何 Σ 输出		16 15	24 24	ns
t_{PLH} t_{PHL}	传输延迟，任何 A 或 B 输入到Σ输出		15 15	24 24	ns
t_{PLH} t_{PHL}	传输延迟，C_0输入到 C_4 输出		11 11	17 22	ns
t_{PLH} t_{PHL}	传输延迟，任何 A 或 B 输入到 C_4 输出		11 12	17 17	ns

图 6.11　74LS283 芯片的传输延迟特性

6.2.3　加法器扩展

◇ 通过级联，加法器可以扩展到更多的位。

使用两个 4 位加法器，可以对 4 位加法器进行扩展，用以处理两个 8 位数字的加法。低 4 位加法器 C_0 的进位输入接地，因为最低有效位是没有进位的，低 4 位加法器的进位输出与较高 4 位加法器的进位输入相连接，如图 6.12(a)所示。这个过程称之为级联(cascading)。注意，在这种情况下，由于进位输出产生在第 8 位上，输出进位用 C_8 表示。低 4 位加法器是把数字中最低的或较低的 4 位有效位相加，而高 4 位加法器就是把这个 8 位数字中最高或较高的 4 位有效位相加。

类似地，如图 6.12(b)所示，还可以使用 4 个 4 位加法器的级联来处理 16 位数字的加法。注意，由于进位输出产生在第 16 位的位置上，所以输出进位用 C_{16} 表示。

例 6.4　如何将两个 74LS283 加法器连接起来组成一个 8 位并行加法器。求下列 8 位输入数的输出位：

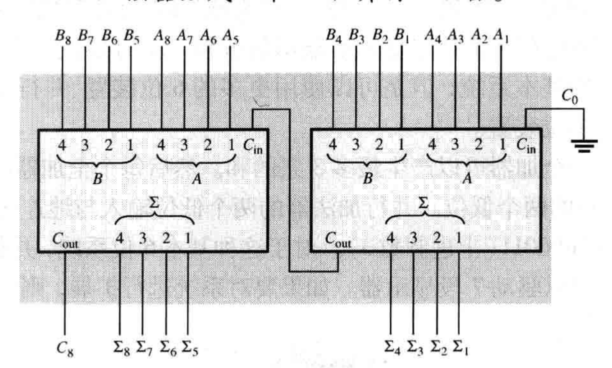
$$A_8A_7A_6A_5A_4A_3A_2A_1 = 10111001 \text{ 和 } B_8B_7B_6B_5B_4B_3B_2B_1 = 10011110$$

解： 使用两个74LS283 4位并行加法器来组成一个8位加法器。两个74LS283之间的唯一连接，是这个低位加法器的进位输出（引脚9）与高位加法器的进位输入（引脚7）之间的连接，如图6.13所示。低位加法器的引脚7接地（没有进位输入）。

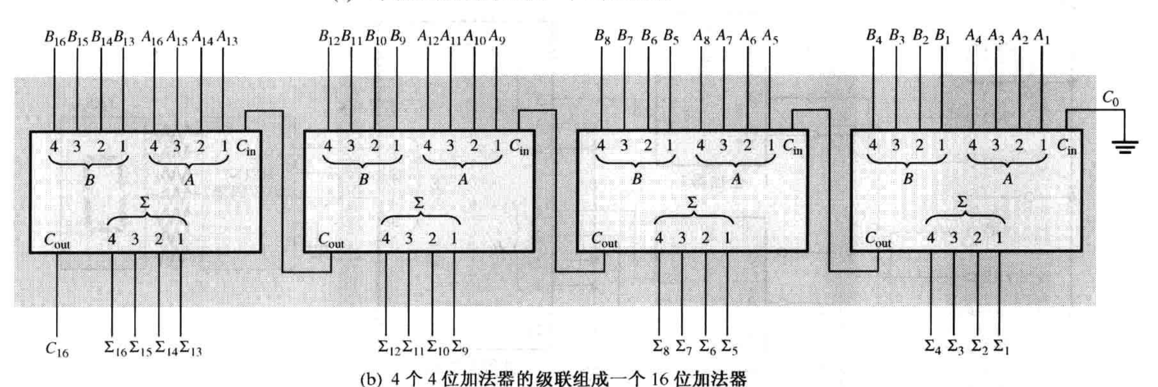
8位数的和是

$$\Sigma_9\Sigma_8\Sigma_7\Sigma_6\Sigma_5\Sigma_4\Sigma_3\Sigma_2\Sigma_1 = 101010111$$

相关问题： 使用74LS283加法器组成一个12位并行加法器。



(a) 4位加法器的级联组成一个8位加法器



(b) 4个4位加法器的级联组成一个16位加法器

图6.12　加法器扩展的例子

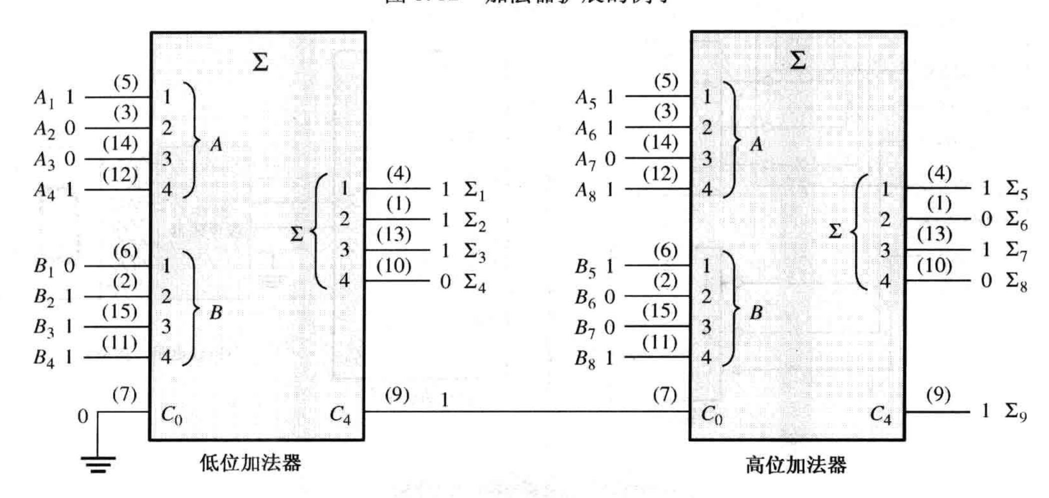


图6.13　两个74LS283加法器相连组成一个8位加法器（括号里的数字为引脚编号）

6.2.4 应用举例

投票系统就是全加器和并行加法器的一个应用例子，它可以在投票的同时计算出“赞成”(YES)及“反对”(NO)的票数。当一群人聚集在一起，需要立即对观点(赞成或反对)、决定和某个争端事件或一些其他事情进行投票时，就需要用到这种系统。

投票系统的最简单的形式，即系统包括的一个“赞成”和“反对”的选择开关，在每个投票人处有一个这样的开关，同时有一个数字显示器显示赞成票、反对票或弃权票的数目。如图6.14所示就是一个6位装置的基本系统，但是可以使用更多的6位装置、并行加法器及显示电路，将这个6位装置扩展为更多位装置。

在图6.14中，每一个全加器可以产生最多3票的和。然后每个全加器的和及进位输出被传送到一个并行二进制加法器的两个低位。并行加法器的两个低位输入与地连接(0)，因为在任何情况下，二进制输入都不会超过0011(十进制数3)。对于这种基本6位系统，并行加法器的输出被传送到一个BCD-7段译码器中以驱动7段显示器。如果要对系统进行扩展，则必须包括更多的电路。

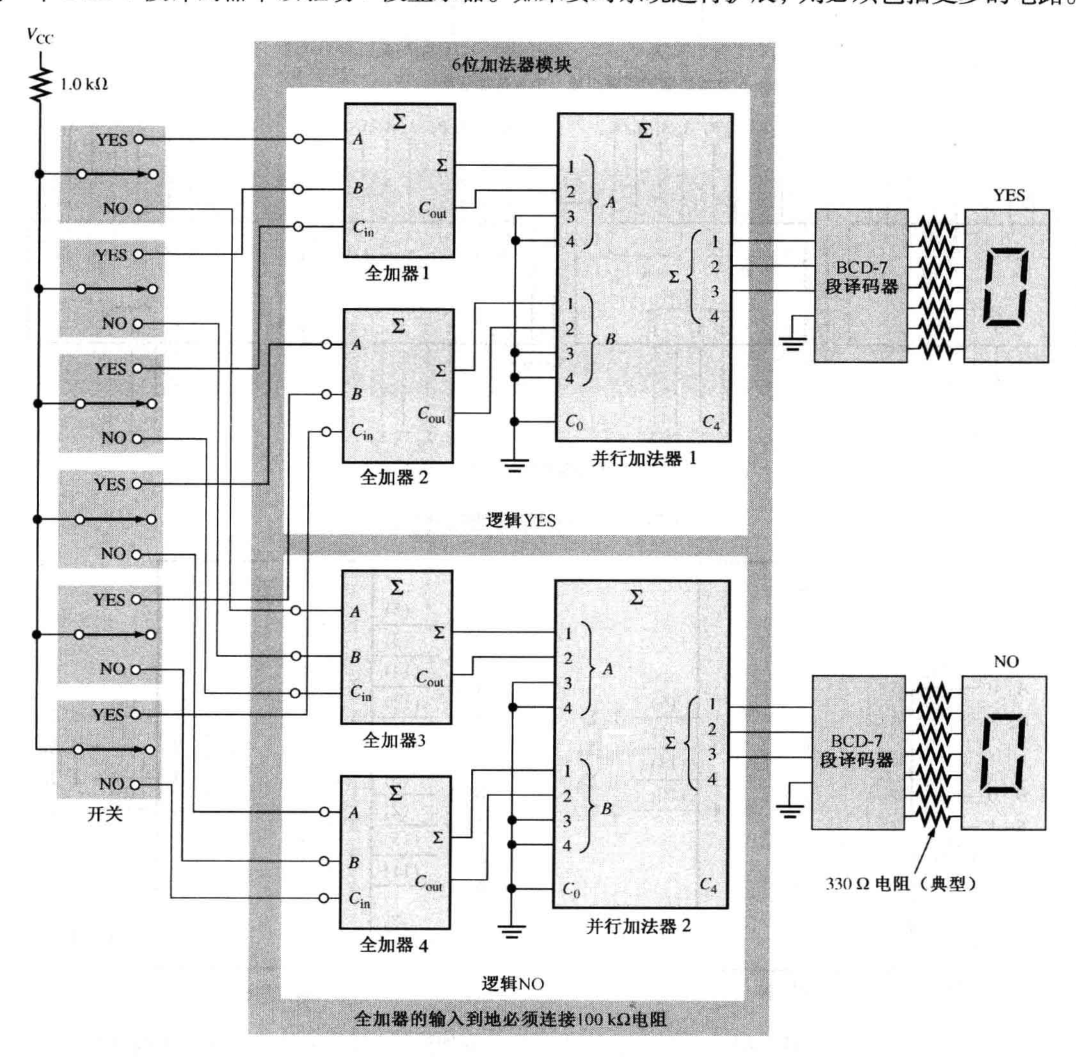


图6.14 使用全加器和并行二进制加法器的投票系统

每个全加器的输入连接电阻到地，这样可以确保当开关处在悬空位置时，每一个输入为低电平(使用 CMOS 逻辑)。当开关位于"赞成"的位置上或"反对"的位置上时，高电平(V_{CC})加在与之相连的全加器输入。

6.2 节　温故而知新

1. 两个 4 位数字(1101 和 1011)加到一个 4 位并行加法器上，进位输入为 1。确定其和(Σ)与进位输出。
2. 两个二进制数相加需要多少个 74LS283 加法器？其中每个二进制数表示 1000_{10} 以内的十进制数。

6.3　异步[①]进位与超前进位加法器

6.3.1　异步进位加法器

异步进位加法器中的每个全加器的进位输出连接到下一个高一级的全加器的输入(一级一个全加器)。任何一级的输出和及进位必须在上一级的进位到来后才能产生。这就造成加法过程的时间延迟，如图 6.15 所示。假设输入 A 和 B 已经到达，每个全加器的进位传输延迟就是从输入进位的到达到输出进位产生的时间。

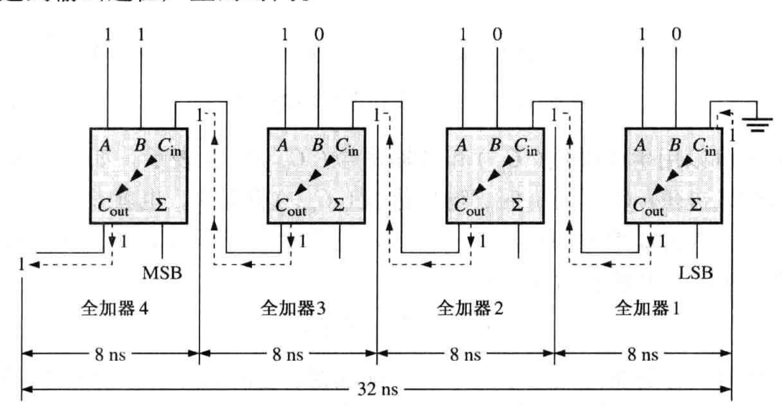


图 6.15　4 位并行异步进位加法器的进位传输延迟的"最坏情况"

全加器 1(FA1)在有进位输入到达以后，才有可能产生进位输出。全加器 2(FA2)在全加器 1 产生进位输出以后，才有可能产生进位输出。全加器 3(FA3)在全加器 1 产生进位输出以后，紧接着还要等全加器 2 的进位输出产生以后，才有可能产生进位输出，以此类推。如图 6.15 所示，最低有效位全加器的进位输入在产生最后的加法结果之前，必须异步途经所有的全加器。所有全加器的传输延迟的积累就是"最坏情况"的加法运算时间。总的延迟时间可能不一样，取决于每个全加器产生进位的时间。如果两个数相加在两个全加器之间没有进位(0)产生，那么相加的时间仅仅是单一全加器的运算时间，即相加数据位加在输入到产生和输出的时间，然而，必须假设最坏的情况总是存在。

6.3.2　超前进位加法器

加法运算可以达到的速度，由于通过并行加法器的每一级的进位传输或异步进位所需时间的原因而受到了限制。通过去除异步进位延迟，加快加法运算速度的一个方法是使用超前进位

① 又称行波：前级输出连到后级输入，信号传送就像波的行进。

(look-ahead carry)加法。超前进位加法器提前使用每一级的输入进位，并基于输入，通过进位生成或进位传输函数而产生进位。

进位生成 当一个输出进位由全加器在内部产生(生成)时，进位生成就发生了。当且仅当两个输入位为1时，就生成了一个进位。生成进位 C_g，表示为两个输入位 A 和 B 的与运算。

$$C_g = AB \tag{6.5}$$

进位传输 当输入进位异步传送成为输出进位时，进位传送就发生了。当全加器的一个或两个输入位为1时，就可能传输了一个输入进位。传输进位 C_p，表示为两个输入位的或运算。

$$C_p = A + B \tag{6.6}$$

图6.16为进位生成和进位传输的情况。三个箭头表示异步(传送)。

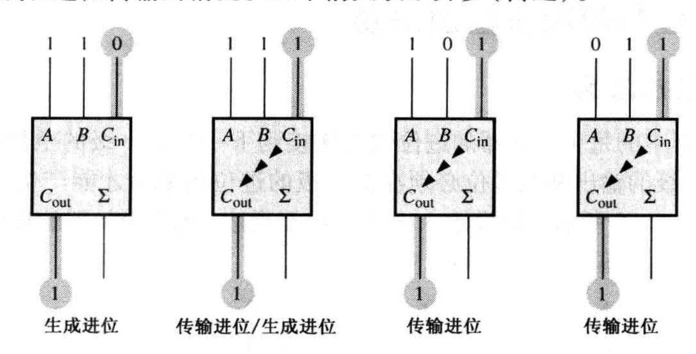


图6.16 进位生成和进位传输的情况

全加器的输出可以用生成进位(C_g)和传输进位(C_p)两者表示。如果生成进位为1或传输进位为1并且输入进位(C_{in})为1时，输出进位(C_{out})就是1。也就是说，如果使用全加器($A = 1$ 与 $B = 1$)，或者如果由加法器传输的输入进位为1($A = 1$ 或 $B = 1$)和 $C_{in} = 1$，那么输出进位就是1。这样的关系表示如下：

$$C_{out} = C_g + C_pC_{in} \tag{6.7}$$

现在来看这样的概念是如何应用于一个并行加法器，它的每一级如图6.17所示的4位加法器。对于每个全加器，输出进位由生成进位(C_g)、传输进位(C_p)和输入进位(C_{in})确定。只要应用输入位 A、B 和最低有效位加法器的输入进位，立即可以得到每一级的 C_g 和 C_p 函数，因为这两个函数仅仅由这些位确定。每一级的输入进位就是前一级的输出进位。

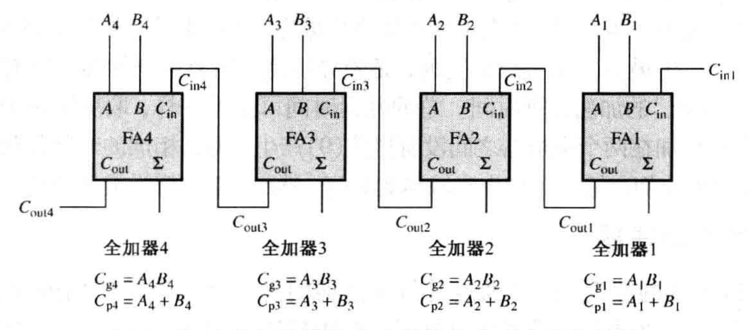


图6.17 4位加法器输入位的进位生成和进位传输

基于这样的分析，现在就可以设计4位加法器的每一级全加器的输出进位(C_{out})的表达式。

全加器 1：

$$C_{out1} = C_{g1} + C_{p1}C_{in1}$$

全加器 2：

$$\begin{aligned} C_{in2} &= C_{out1} \\ C_{out2} &= C_{g2} + C_{p2}C_{in2} = C_{g2} + C_{p2}C_{out1} = C_{g2} + C_{p2}(C_{g1} + C_{p1}C_{in1}) \\ &= C_{g2} + C_{p2}C_{g1} + C_{p2}C_{p1}C_{in1} \end{aligned}$$

全加器 3：

$$\begin{aligned} C_{in3} &= C_{out2} \\ C_{out3} &= C_{g3} + C_{p3}C_{in3} = C_{g3} + C_{p3}C_{out2} = C_{g3} + C_{p3}(C_{g2} + C_{p2}C_{g1} + C_{p2}C_{p1}C_{in1}) \\ &= C_{g3} + C_{p3}C_{g2} + C_{p3}C_{p2}C_{g1} + C_{p3}C_{p2}C_{p1}C_{in1} \end{aligned}$$

全加器 4：

$$\begin{aligned} C_{in4} &= C_{out3} \\ C_{out4} &= C_{g4} + C_{p4}C_{in4} = C_{g4} + C_{p4}C_{out3} \\ &= C_{g4} + C_{p4}(C_{g3} + C_{p3}C_{g2} + C_{p3}C_{p2}C_{g1} + C_{p3}C_{p2}C_{p1}C_{in1}) \\ &= C_{g4} + C_{p4}C_{g3} + C_{p4}C_{p3}C_{g2} + C_{p4}C_{p3}C_{p2}C_{g1} + C_{p4}C_{p3}C_{p2}C_{p1}C_{in1} \end{aligned}$$

注意，对于这些表达式，每一级全加器输出进位仅由最初的输入进位（C_{in1}）及那一级的函数 C_g 和 C_p 和前一级的函数 C_g 和 C_p 确定。因为每一级的函数 C_g 和 C_p 可以由这一级全加器的输入 A 和 B 来表示，所以所有的输入进位可以立即获得（除了门的延迟），不需要等到进位异步传输通过每一级后才得到最后结果。因此，超前进位技术加快了加法运算的速度。

如图 6.18 所示，使用逻辑门并连接全加器产生的 4 位超前加法器，同样获得了 C_{out} 的表达式。

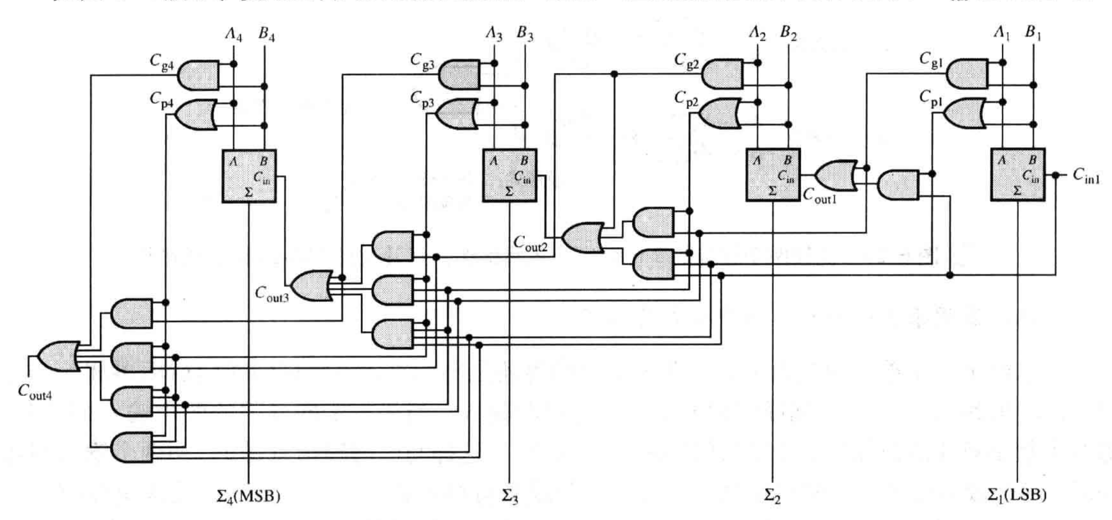


图 6.18　4 位超前进位加法器的逻辑框图

6.3.3　超前进位和异步进位加法器的组合

如多数集成加法器一样，6.2 节介绍的 4 位加法器 74LS283 是一个超前进位加法器。当这些加法器级联起来，把处理二进制数的能力扩展到四位以上时，一个加法器的输出进位将连接到下一个加法器的输入进位。当两个以上的 74LS283 级联时，就在 4 位加法器中间产生了异步进位。最终的加法器实际上是一个超前进位和异步进位的组合。每个中规模集成电路（MSI）加法器的内部运行超前进位，当进位在一个加法器的外部传输到下一个时，异步进位就起作用了。

6.3节 温故而知新

1. 一个全加器的输入为 $A=1$ 和 $B=0$，确定其 C_g 和 C_p 的值。

2. 当一个全加器的 $C_{in}=1$、$C_g=0$ 和 $C_p=1$ 时，确定其输出进位的值。

6.4 比较器

6.4.1 相等

如在第3章所学到的，同或门可以用做一个基本的比较器，因为它的两个输入位不相等时输出为0，相等时输出为1。图6.19给出了同或门用做一个2位比较器的情况。

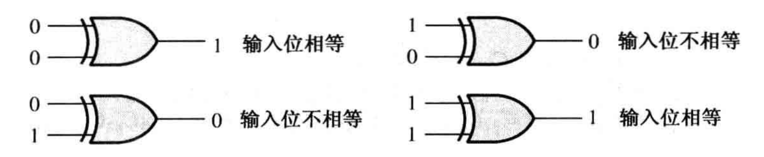


图6.19 基本比较器的运算

为了比较两个2位二进制数的大小，需要更多的同或门。门 G_1 用来比较这两个数的两个最低有效位，门 G_2 用来比较两个最高有效位，如图6.20所示。如果两个数是相等的，它们所对应的位也是相同的，那么每个同或门的输出为1。如果对应的两个位不相等，则同或门的输出为0。

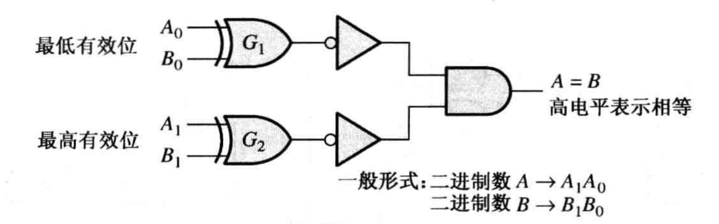


图6.20 比较两个相等的2位数字的逻辑框图。打开文件F06-20检验该操作

◇ 比较器用来判断两个二进制数是否相等。

为了产生一个单一的输出来表示两个数字相等还是不相等，需要在同或门后面再加一个与门，如图6.20所示。每个同或门的输出加到与门的输入。当每一个同或门的两个输入相等时，即两个数字对应的位相等，使得与门的两个输入为1，这样与门的输出就是1。当两个输入不相等时，两个数字位相对应的位中其中一对不相等或两对都不相等，这时与门的输入至少有一个为0，使得与门的输出为0。因此，与门的输出表示两个数的相等(1)或不相等(0)。例6.5解释了这种运算的两种具体情况。同或门符号代替了异或门和反相器。

例6.5 在图6.21中，下列每个二进制数加到比较器的输入，求下列逻辑电平通过电路时的输出。

(a)10 和 10 (b)11 和 10

解：(a)当输入为10和10时，输出为**1**，如图6.21(a)所示。

(b)当输入为11和10时，输出为**0**，如图6.21(b)所示。

相关问题：当二进制输入为01和10时，重复这个过程。

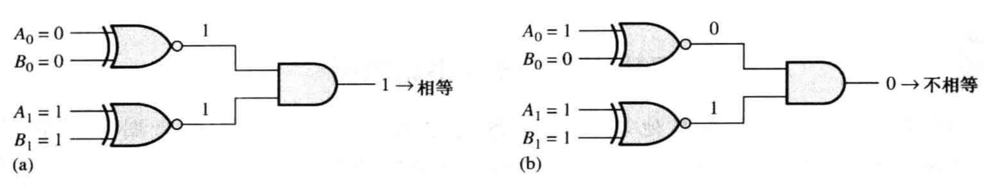


图6.21



计算机小知识

在计算机中，缓冲存储器是介于中央处理器(CPU)和低速主存储器之间的高速存储器。CPU通过它在存储器中的地址(唯一的地址)获取数据。部分地址称为标识符。标识符地址比较器将CPU中的标识符与缓存目录中的标识符进行比较。如果二者是一致的，那么这个地址数据已经在缓存中，可以很快地获取。如果这两个标识符不一致，那么数据则必须从主存储器中以慢得多的速率获取。

如在第3章所知道的，这个基本比较器可以扩展到任何位数。当这两个数自身相等时，与门决定了它们所对应的位必定相等。

6.4.2　不相等

除了相等输出之外，许多集成电路比较器提供附加的输出，表示两个相比较的二进制数哪一个大。也就是说，当数A比数B大($A>B$)时有输出，或当数A比数B小($A<B$)时有输出。4位比较器的逻辑符号如图6.22所示。

为了确定两个二进制数A和B是不相等的，首先需要判断每个数字的最高位。下面是可能的情况：

1. 如果$A_3=1$和$B_3=0$，数A大于数B；
2. 如果$A_3=0$和$B_3=1$，数A小于数B；
3. 如果$A_3=B_3$，这时必须判断下一个较低位的不相等性。

这三个运算对于上述数字的每个位都是有效的。比较器运算的一般过程就是从最高有效位开始，检查这一个位的不相等性。当找到了这个不相等性时，这两个数字之间的关系也就确立了，其他较低位的大小关系可以忽略，因为较低位可能会有与之相反的情况发生；最高位的结果必须优先考虑。

例6.6　如图6.23所示，比较器的输入数字已经给出，求当$A=B$、$A>B$和$A<B$时的输出。

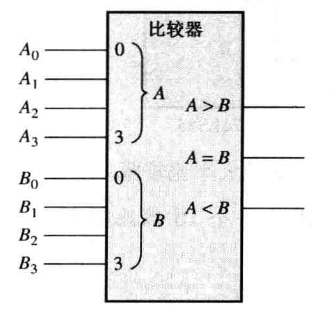


图6.22　具有不相等输出的4位比较器的逻辑符号

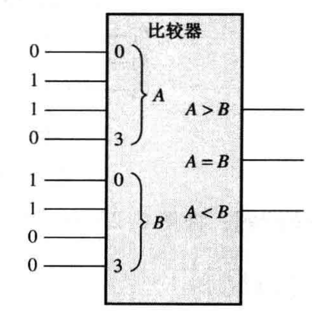


图6.23

解：数字A的输入为0110，数字B的输入为0011。**$A>B$的输出为高电平，其他输出为低电平。**

相关问题：当$A_3A_2A_1A_0=1001$和$B_3B_2B_1B_0=1010$时，求比较器的输出。



74LS85 4位大小比较器

74LS85是一个比较器，在其他集成电路系列中也可以找到它。引脚图和逻辑符号如图6.24所示。注意，这种比较器除了含有前面介绍的常用比较器的所有输入及输出之外，还有3个级联输入：$A<B$、$A=B$和$A>B$。这些输入允许几个比较器级联起来，用来比较大于4位数字的数。为了扩展比较器，低位比较器的$A<B$、$A=B$和$A>B$输出与对应的下一个较高位比较器的级联输入相连接。最低位比较器在$A=B$输入端口必须接高电平，在$A<B$和$A>B$输入端口必须接低电平。

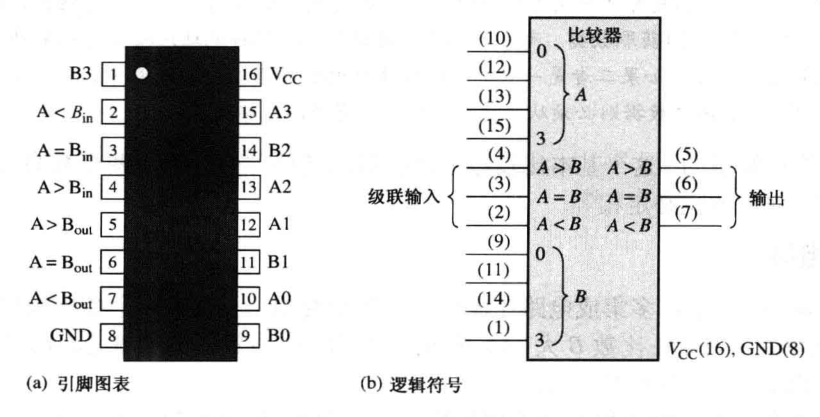


图6.24 74LS85 4位大小比较器的引脚图和逻辑符号(引脚编号在括号内)

例6.7 使用74LS85比较器比较两个8位数的数值大小。给出比较器的恰当连接。

解： 要比较两个8位数需要两个74LS85芯片。如图6.25所示，两个芯片以级联的方式连接。

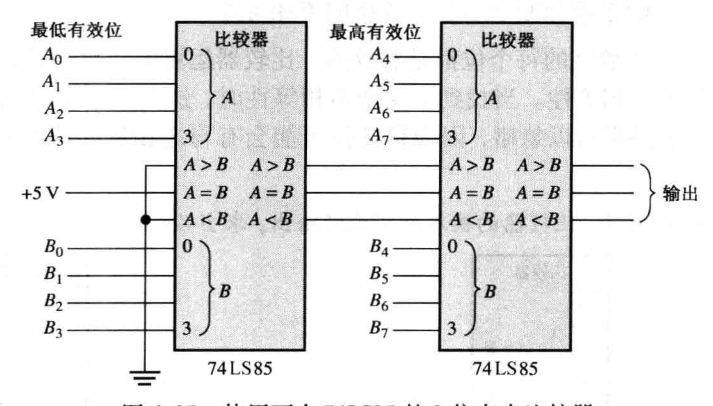


图6.25 使用两个74LS85的8位大小比较器

相关问题： 将图6.25中的电路进行扩展，使其成为一个16位比较器。

温故而知新

1. 二进制数A=1011和B=1010作为74LS85的输入，请确定其输出。
2. 二进制数A=11001011和B=11010100用于图6.25的8位比较器，请确定每一个比较器的数据状态。

6.5 译码器

6.5.1 基本二进制译码器

假设需要确认一个二进制数1001出现在一个数字电路的输入中。在这个基本译码电路中，需要使用一个与门，因为只有当所有的输入都为高电平时，它才产生一个高电平。因此，当二进制数1001出现时，就必须确定与门的所有输入都为高电平；这可以通过把中间两个位(0)反相来满足要求，如图6.26所示。

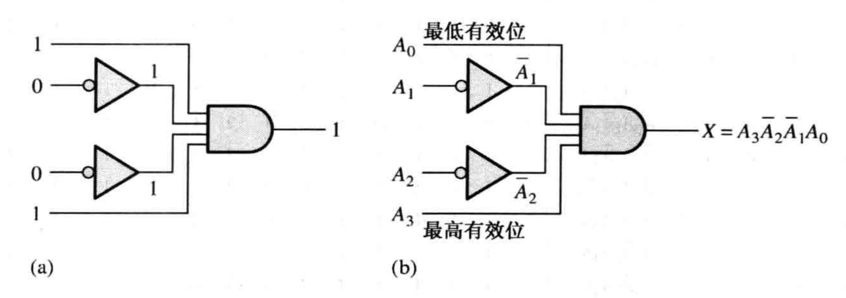


图6.26　高电平输出有效的二进制数1001的译码逻辑

由如图6.26(b)给出的解释对应的逻辑方程，得到图6.26(a)的译码器。可以验证，当输入$A_0=1$、$A_1=0$、$A_2=0$和$A_3=1$时，输出为1，其他情况的输出为0。其中A_0是最低有效位，A_3是最高有效位。本书中，在一个二进制数或其他有权码的表示方法中，最低有效位指的是水平方向最右边的位，或者垂直方向最上面的位，除非具体说明。

在图6.26中，如果用一个与非门来代替这个与门，一个低电平输出将表示相应二进制代码的输入，在此情况下，二进制代码仍然是1001。



计算机小知识

指令告诉计算机应该进行什么运算。指令以机器码(1和0)的形式存在，为了使计算机执行指令，需要对指令译码。指令译码是指令流水线中的一步，过程如下：从存储器读取指令(取指令)，指令被译码，从存储器读取操作数(取操作数)，执行指令，结果写回存储器。基本上，流水线就是使当前的指令完成后，开始执行下一条指令。

例6.8　求出对二进制数1011进行译码，输出为高电平的译码逻辑电路。

解： 在所求的二进制数中，只要对以0出现的变量取反就可以得到译码函数，如下所示：

$$X = A_3\bar{A}_2A_1A_0 \qquad (1011)$$

把原变量A_0、A_1、A_3和与门的输入直接相连，再把变量A_2取反以后加到与门的输入，就实现了以上的函数。译码逻辑电路如图6.27所示。

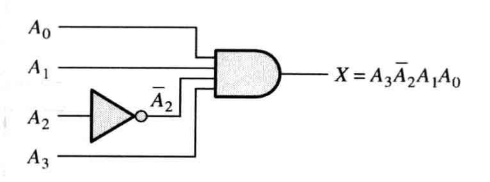


图6.27　当输入为1011时产生一个高电平输出的译码逻辑电路

相关问题： 将这个译码逻辑电路扩展，用来检测二进制代码10010，并产生一个低电平有效输出。

6.5.2　4 位译码器

为了对所有的 4 位组合进行译码，需要 16 个译码门($2^4 = 16$)。这种译码器通常称为 4 线-16 线译码器，因为有 4 个输入和 16 个输出；或者称为 16 选 1 译码器，因为给出的任何一个输入，在 16 个输出里只有一个是有效的。表 6.4 给出 16 个二进制代码和与之相对应的译码函数。

表 6.4　低电平有效输出的 4 线-16 线(16 选 1)译码器的译码函数和真值表

十进制数	二进制输入				译码函数	输出															
	A_3	A_2	A_1	A_0		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	1	0	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	0	1	1	$\overline{A}_3\overline{A}_2A_1A_0$	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
4	0	1	0	0	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
5	0	1	0	1	$\overline{A}_3A_2\overline{A}_1A_0$	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
6	0	1	1	0	$\overline{A}_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
7	0	1	1	1	$\overline{A}_3A_2A_1A_0$	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
8	1	0	0	0	$A_3\overline{A}_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
9	1	0	0	1	$A_3\overline{A}_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
10	1	0	1	0	$A_3\overline{A}_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
11	1	0	1	1	$A_3\overline{A}_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
12	1	1	0	0	$A_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
13	1	1	0	1	$A_3A_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
14	1	1	1	0	$A_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
15	1	1	1	1	$A_3A_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

如果对于每个译码的数，需要一个高电平输出，那么整个译码器可以由与非门和反相器来实现。为了对 16 个二进制数进行译码，需要 16 个与非门(与门可以用来产生高电平有效输出)。

图 6.28 为一个低电平有效输出的 4 线-16 线(16 选 1)译码器的逻辑符号。BIN/DEC 表示的是一个二进制输入产生相对应的十进制有效输出。输入标号为 8、4、2 和 1 的输入代表的是输入位的二进制权($2^3 2^2 2^1 2^0$)。

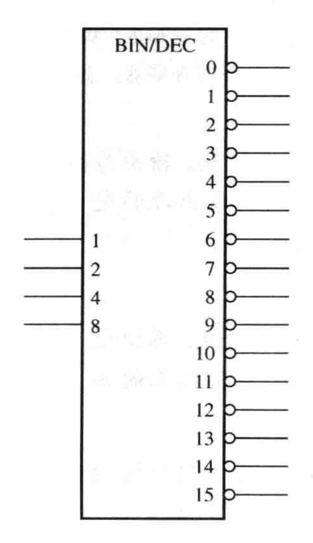


图 6.28　4 线-16 线(16 选 1)译码器的逻辑符号。打开文件 F06-28 检验该操作



74HC154 16 选 1 译码器

74HC154 是集成电路译码器的一个很好的例子，其逻辑符号如图6.29 所示。这个器件提供了一个使能函数(EN)，它由一个或非门实现，这里把或非门作为一个非-与门来使用。每个片选输入 $\overline{CS}_1$ 和 $\overline{CS}_2$ 为低电平有效输入，低电平有效输入使得使能门输出(EN)高电平。这个使能门输出和译码器中每个与非门的一个输入相连接，所以使能门输出必须是高电平，这样才能使与非门工作。如果使能门的两个输入不是有效低电平，这时无论 4 个输入变量 A_0、A_1、A_2 和 A_3 是什么值，译码器的所有 16 个输出(Y)全部为高电平。

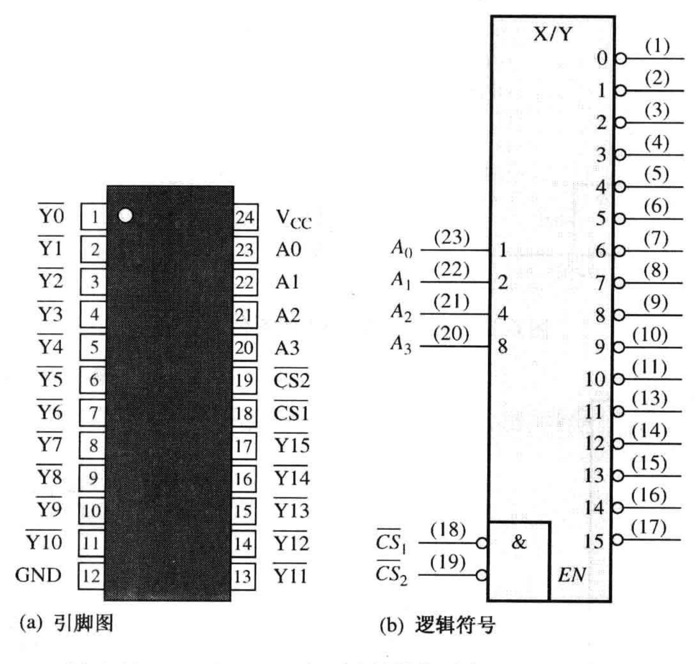


图 6.29　74HC154 16 选 1 译码器的引脚图和逻辑符号

例 6.9　某些实际应用需要对一个 5 位数译码。利用 74HC154 译码器实现这个逻辑。二进制数用 $A_4\ A_3\ A_2\ A_1\ A_0$ 的形式来表示。

解：由于 74HC154 只能处理 4 位数，那么必须使用两个译码器对 5 位数进行译码。第 5 位 A_4 和其中一个译码器的片选输入 $\overline{CS}_1$ 和 $\overline{CS}_2$ 相连接，$\overline{A}_4$ 与另一个译码器的输入 $\overline{CS}_1$ 和 $\overline{CS}_2$ 相连接，如图 6.30 所示。当十进制数为 15 或更小时，$A_4=0$，低位译码器工作，高位译码器不工作。当十进制数大于 15 时，$A_4=1$，而 $\overline{A}_4=0$，高位译码器工作，低位译码器不工作。

相关问题：如图 6.30 所示，求二进制有效输入 10110 的输出。

6.5.3　应用举例

译码器在许多应用场合中使用。图 6.31 描绘的常见电路图是计算机中用于选择输入/输出的例子。

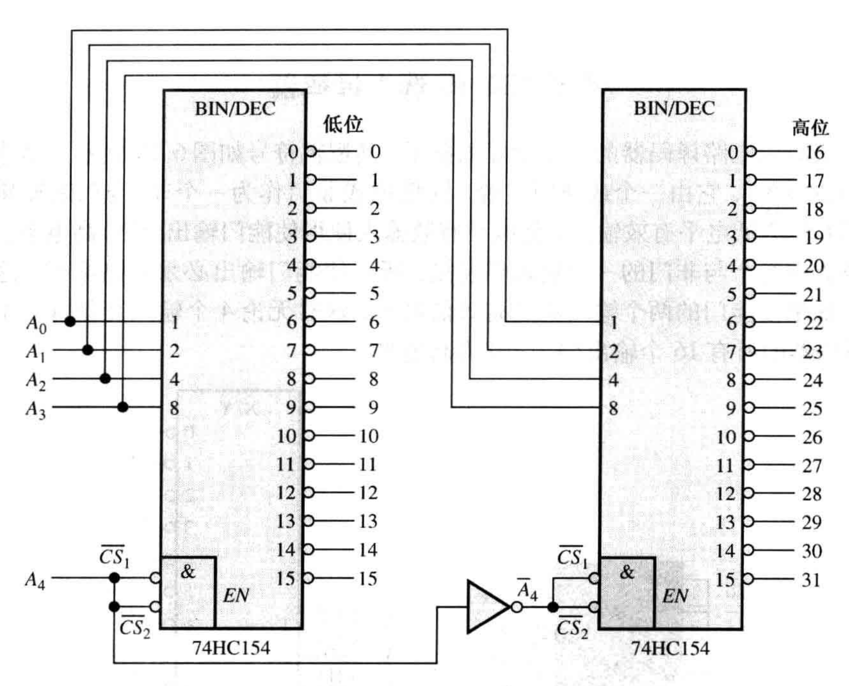


图 6.30　使用 74HC154 的 5 位译码器

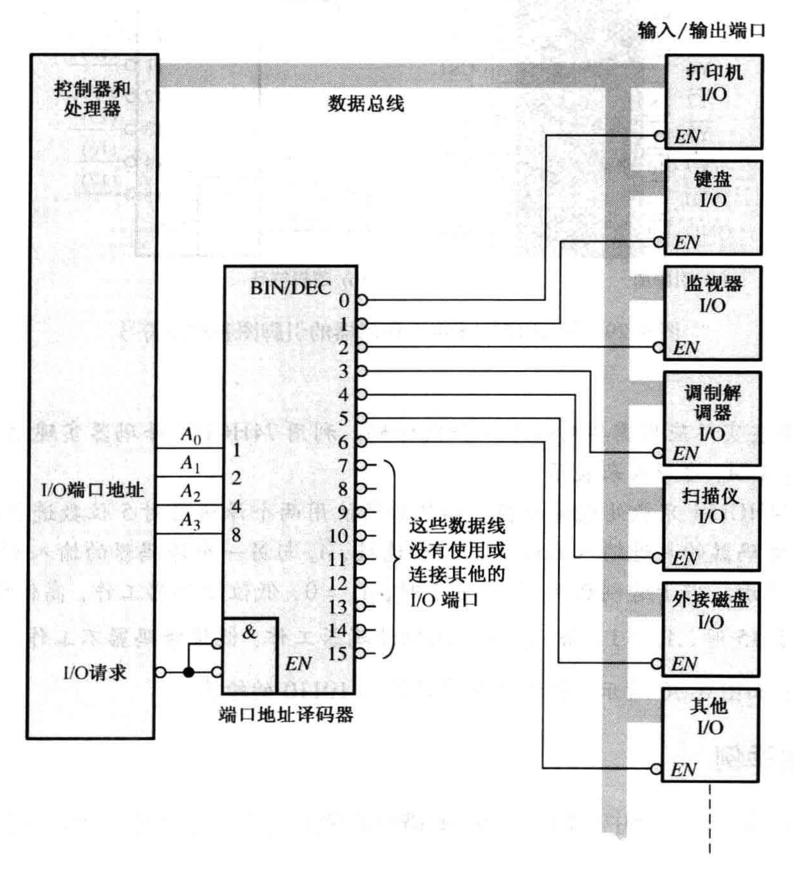


图 6.31　一个只有 4 条地址线的端口地址译码器的简单计算机 I/O 端口系统

计算机必须和各种外部(称为外围)设备进行通信，这通过输入/输出(I/O)端口发送和/或接收数据来完成。这些外部设备包括打印机、调制解调器、扫描仪、外部磁盘驱动、键盘、视频监视器和其他计算机。如图6.31所示，计算机使用译码器选择I/O端口，这样可以向一个具体的外部设备传送数据或从它接收数据。

每一个输入/输出端口都有唯一识别的数字，称为地址。当计算机需要和一台特定的设备通信时，它发出相应的地址码，也就是和这个特定设备连接的输入/输出端口地址。这个二进制端口地址被译码器译码，然后译码器输出所需的电平，使输入/输出端口开通。

如图6.31所示，二进制数据在计算机内部的数据总线上传输，数据总线是一组并行的线。例如，一组8位总线由并行的8条线组成，每次可以传输一个数据字节。数据总线和所有输入/输出端口相连接，但是，任何进入或出来的数据都必须通过端口地址译码器的译码，使得端口开通才能够传输。

6.5.4 BCD-十进制译码器

BCD-十进制译码器把每一个BCD码(8421码)转化成10个十进制数中的一个数所表示的输出。该译码器经常被称为4线-10线译码器或10选1译码器。

实现此译码器的方法与前面所讲到的16选1译码器是相同的，唯一的不同就是只需要10个译码门，这是因为BCD码只表示10个十进制数0~9。表6.5列出了10个BCD码及与它们相对应的译码函数。这些译码函数由与非门实现，提供低电平有效输出。如果需要高电平有效输出，那么应使用与门译码。这种逻辑与16选1译码器的前10个译码门相同(见表6.5)。

表6.5　BCD译码功能

十进制数	BCD码 A_3	A_2	A_1	A_0	译码函数
0	0	0	0	0	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$
1	0	0	0	1	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$
2	0	0	1	0	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$
3	0	0	1	1	$\overline{A}_3\overline{A}_2A_1A_0$
4	0	1	0	0	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$
5	0	1	0	1	$\overline{A}_3A_2\overline{A}_1A_0$
6	0	1	1	0	$\overline{A}_3A_2A_1\overline{A}_0$
7	0	1	1	1	$\overline{A}_3A_2A_1A_0$
8	1	0	0	0	$A_3\overline{A}_2\overline{A}_1\overline{A}_0$
9	1	0	0	1	$A_3\overline{A}_2\overline{A}_1A_0$



74HC42 BCD-十进制译码器

74HC42 BCD-十进制译码器把每一个BCD码(8421码)转换成10个十进制数中的一个数所表示的输出。逻辑符号如图6.32所示。

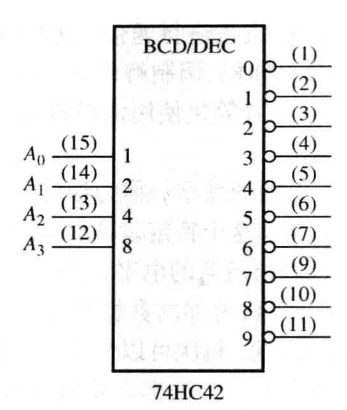


图 6.32 74HC42 BCD-十进制译码器的逻辑符号

例 6.10 如果将图 6.33(a)的输入波形加在 74HC42 的输入，请给出其输出波形。

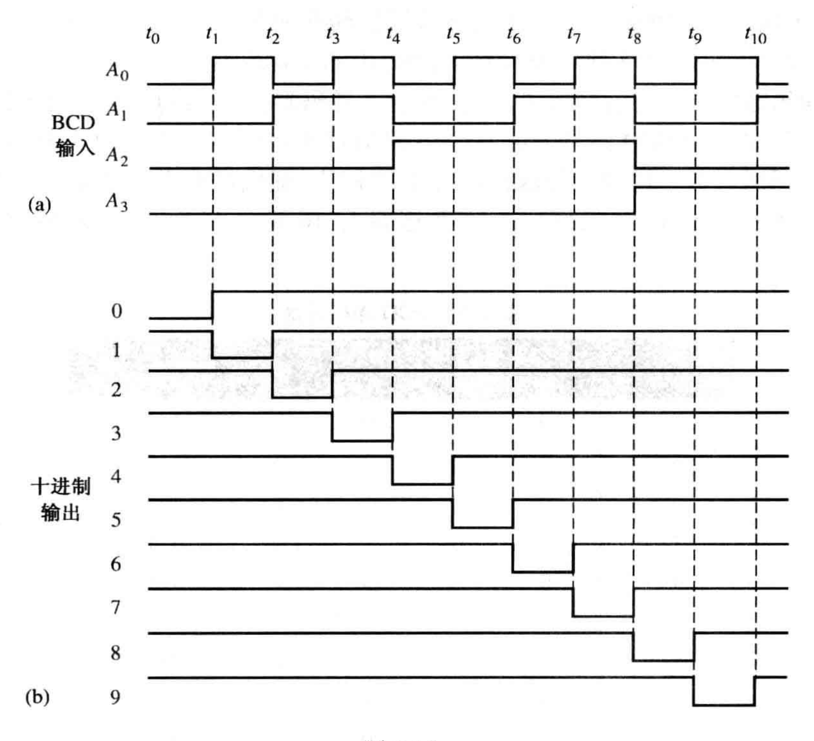


图 6.33

解：如图 6.33(b)所示的输出波形。由图可以知道，输入按照 BCD 码 0 ~9 的顺序变化。时序图中的输出波形给出了十进制输出的顺序。

相关问题：BCD 码的输入序列为下列十进制数：0，2，4，6，8，1，3，5，9。构建一个时序图，显示相应的输入和输出波形。

6.5.5 BCD-7 段译码器

BCD-7 段译码器的输入接收输入的 BCD 码，并产生一个用来驱动 7 段显示器件的输出，显示器件产生一个十进制读数。一个基本 7 段译码器的逻辑框图如图 6.34 所示。

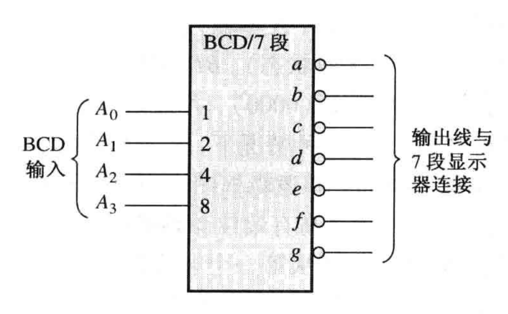


图 6.34　低电平有效输出的 BCD－7 段译码器/驱动器的逻辑符号。打开文件 F06-34 检验该操作



74LS47 BCD－7 段译码器/驱动器

74LS47 是集成电路器件的一个例子，它把一个 BCD 输入进行译码，并驱动 7 段显示器。除了译码和段驱动功能之外，74LS47 还有另外几个附加的功能，如图 6.35 所示的逻辑符号中 $\overline{LT}$、$\overline{RBI}$、$\overline{BI}/\overline{RBO}$ 的功能。图中这些逻辑符号上的小圆圈，表示所有的输出（a 到 g）都是低电平有效，同样 $\overline{LT}$（灯测试）、$\overline{RBI}$（异步灭零输入）和 $\overline{BI}/\overline{RBO}$（灭零输入/异步灭零输出）也是低电平有效。输出可以直接驱动一个共阳 7 段显示器（回顾第 4 章讨论过的 7 段显示器）。74LS47 除了对一个 BCD 输入译码，并产生一个响应的 7 段输出之外，还具有灯测试和灭零功能。

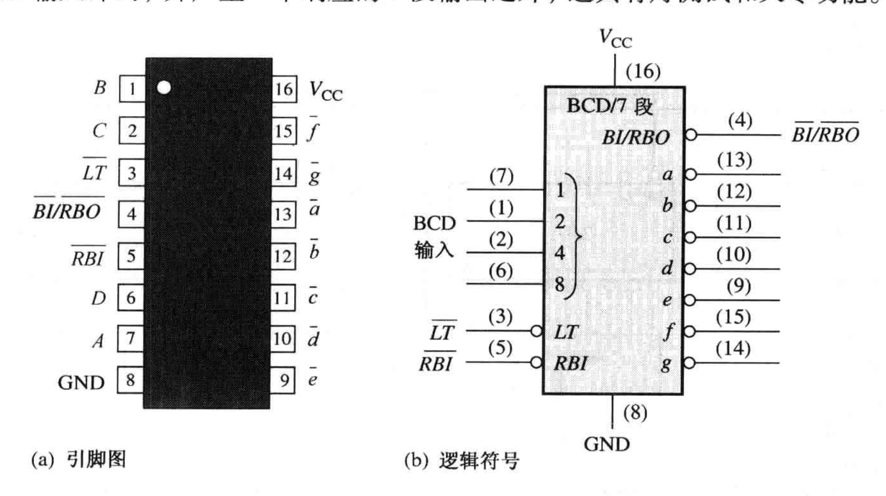


图 6.35　74LS47 BCD－7 段译码器的引脚图和逻辑符号

灯测试　当 $\overline{LT}$ 输入为一个低电平、$\overline{BI}/\overline{RBO}$ 为高电平时，7 段显示器的每一段都被点亮。灯测试用来检测是否有段已经烧坏。

灭零　灭零用于取消多位数中不必要的 0 的显示。例如，在 6 位数字显示器中，如果 0 没有熄灭，数字 6.4 可能被显示为 006.400。把数字前面的 0 熄灭称为头部灭零，把数字尾端的 0 熄灭称为尾部灭零。记住，只是把不需要的零熄灭。由于有了灭零，数字 030.080 将显示为 30.08（需要的零仍然保留）。

灭零使得一个数头部或尾部的 0 不会在显示器上显示。在 74LS47 中，灭零是通过使用 $\overline{RBI}$ 和 $\overline{BI}/\overline{RBO}$ 功能实现的。$\overline{RBI}$ 是 74LS47 的异步灭零输入，而 $\overline{RBO}$ 是 74LS47 的异步灭零输出；它们一起用于灭零。$\overline{BI}$ 是灭零输入，它与 $\overline{RBO}$ 共享同样的引脚；也就是引脚 $\overline{BI}/\overline{RBO}$ 既可以用做

输入也可以用做输出。当把它看成输入 $\overline{BI}$(灭零输入)且为低电平时，它优先于所有其他的输入，并且使得所有的段输出为高电平(无效状态)。$\overline{BI}$ 功能不属于芯片灭零功能的一部分。

如果在译码器的 BCD 输入一个零代码(0000)，并且它的 $\overline{RBI}$ 为低电平，那么这个译码器的所有段输出都是无效状态(高电平)。在这种情况下，显示器熄灭，并在 $\overline{RBO}$ 产生一个低电平。

图 6.36(a)给出了一个整数的头部灭零逻辑框图。当译码器的 BCD 输入为 0 时，最高有效位(最左边)的输出 0 总是被熄灭，因为最高有效位译码器的 $\overline{RBI}$ 接地为低电平。每个译码器的 $\overline{RBO}$ 连接到低一级译码器的 $\overline{RBI}$ 上，这样从第一个非零数字开始，左边的零全部熄灭。例如，图 6.36(a)中两个最高位的数字是零，因此全部熄灭。剩下的两个数字 3 和 0 将显示出来。

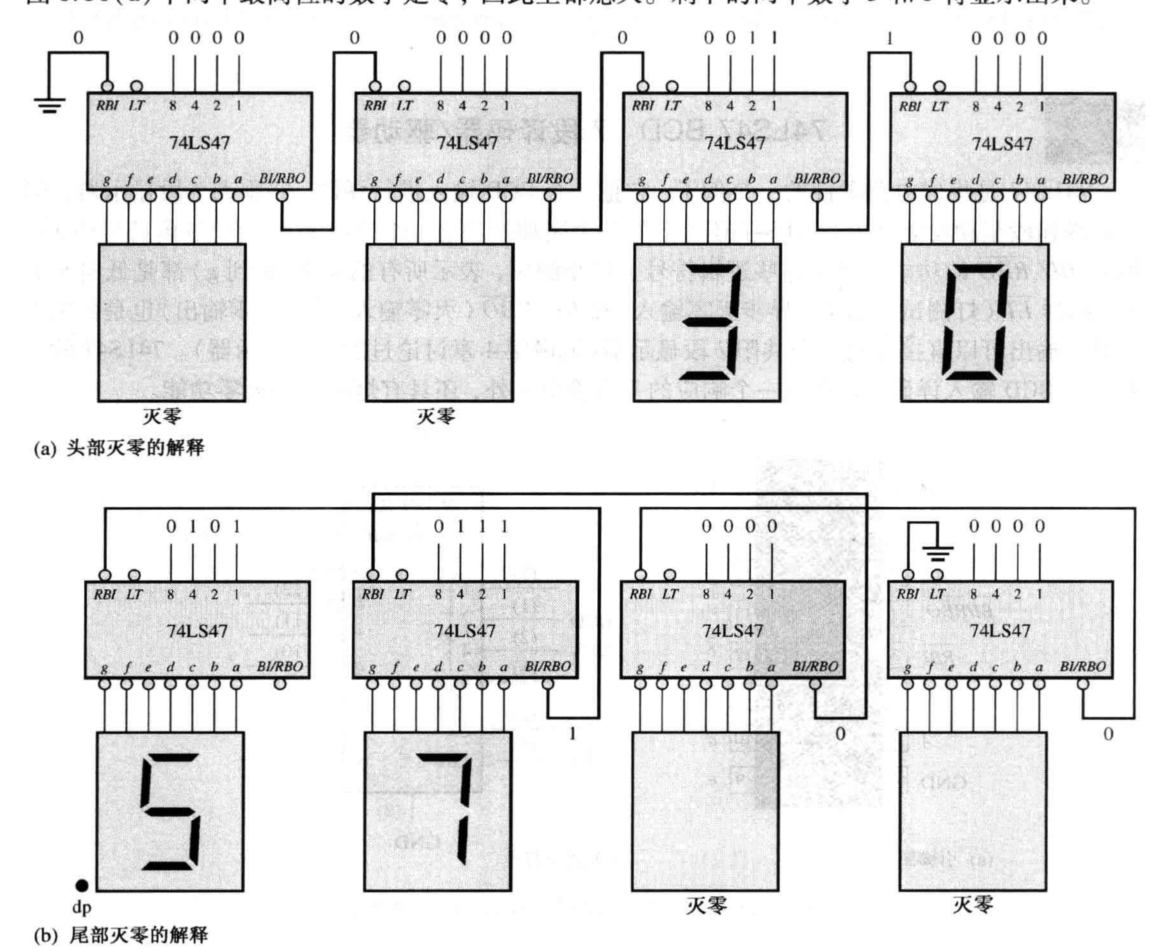


图 6.36　使用 74LS47 BCD－7 段译码器/驱动器的灭零举例

如图 6.36(b)所示，给出一个小数的尾部灭零的逻辑框图。当译码器的 BCD 输入为 0 时，最低有效位(最右边)的输出 0 总是被熄灭，因为最低有效位译码器的 $\overline{RBI}$ 接地为低电平。每个译码器的 $\overline{RBO}$ 连接到高一级译码器的 $\overline{RBI}$ 上，这样从第一个非零数字开始，右边的零全部熄灭。例如，图 6.36(b)中两个最低位的数字是零，因此全部熄灭。剩下的两个数字 5 和 7 是显示的。为了在同一显示器上使头部和尾部灭零组合在一起，以及能够显示十进制数的小数点，还需要附加一些逻辑功能。

6.5 节　温故而知新

1. 一个 3-8 译码器可以用来进行八进制到十进制的译码，当输入为 101 时哪一个输出有效。
2. 对 6 位二进制数进行译码，需要多少个 74HC154 1 线-16 线译码器？
3. 驱动一个共阴 7 段 LED 显示器，选择一个高电平有效还是低电平有效的译码/驱动器？

6.6　编码器

6.6.1　十进制-BCD 编码器

这种类型的编码器有 10 个输入端，每个十进制数对应一个输入端，4 个输出端对应 BCD 码，如图 6.37 所示。这是一个基本的 10 线-4 线编码器。

表 6.6 列出了 BCD 码(8421 码)。从表中可以确定每个 BCD 位和十进制数之间的关系，以便分析逻辑函数。例如，BCD 码的最高有效位相对十进制数 8 或 9 时，A_3总是为 1。因此，A_3位的十进制数的或表达式可以写成

$$A_3 = 8 + 9$$

对于十进制数 4、5、6 或 7，A_2位始终为 1，其或函数的表示法为

$$A_2 = 4 + 5 + 6 + 7$$

对于十进制数 2、3、6 或 7，A_1位始终为 1，其表示法为

$$A_1 = 2 + 3 + 6 + 7$$

最后，对于十进制数 1、3、5、7 或 9，A_0位始终为 1，其表示法为

$$A_0 = 1 + 3 + 5 + 7 + 9$$

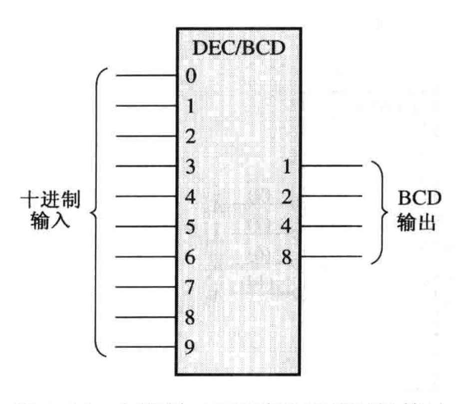


图 6.37　十进制-BCD 编码器的逻辑符号

表 6.6

十进制数	BCD 码 A_3	A_2	A_1	A_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

现在，使用刚刚推导出的逻辑表示法来实现这个逻辑电路，它把每一个十进制数转换成 BCD 码。实现起来很简单，把相对应的十进制数输入线相或，从而形成每个 BCD 输出。如图 6.38所示，这些表达式获得一个基本编码逻辑电路。

如图6.38所示的这个电路的基本操作如下：当一条十进制数输入线为高电平时，在BCD输出线上会产生相应的输出电平。例如，如果输入线9为高电平(假设所有其他输入线为低电平)，在这种条件下，输出端A_0和A_3产生高电平，A_1和A_2产生低电平，这就是十进制数9的BCD码(1001)。

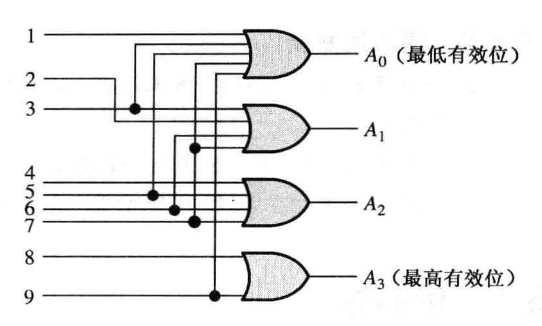


图6.38 十进制-BCD编码器的逻辑框图。不需要0输入，因为当输入都是低电平时，BCD输出也全为低电平



计算机小知识

一个汇编程序可以看成是一个软件编码器，因为它对一个程序写出的助记指令进行编译，通过把每个助记符转换成计算机可以识别的机器码指令(一系列0和1)，实现了把助记指令转换成可执行代码的过程。微处理器的助记指令的例子是：ADD(加)，MOV(传送数据)，MUL(相乘)，XOR，JMP(跳转)，OUT(输出到端口)。

十进制-BCD优先编码器 这种类型的编码器具有的基本编码功能和前面所讨论的相同。一个优先编码器还提供了附加的灵活特性，即它可以应用在需要优先确定的场合。优先权功能意味着编码器将按照最高位十进制数的有效输入产生一个BCD输出，而不考虑任何其他低位的有效输入。例如，如果6和3都为有效输入时，BCD输出为0110(表示十进制数6)。



74HC147十进制-BCD编码器

74HC147为优先编码器，对于十进制数1~9，输入(0)低电平有效，且为BCD低电平有效输出，如图6.39中的逻辑符号所示。当输入都为无效时，则会产生一个BCD零输出。括号中是这个芯片的引脚号。

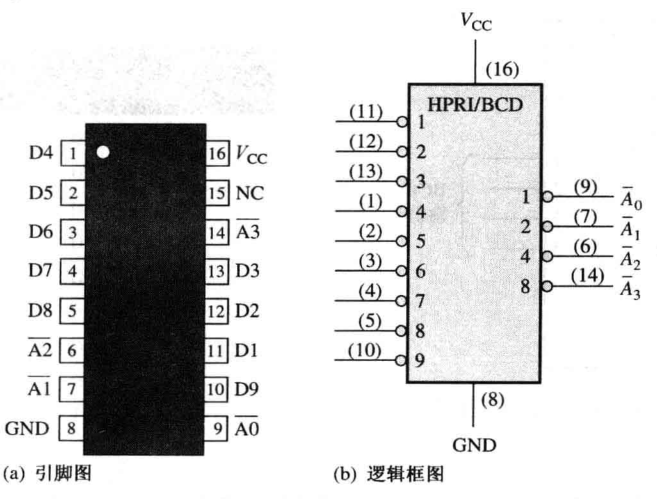


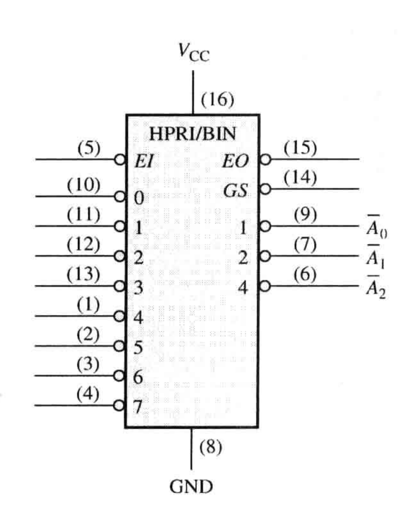
图6.39 74HC147十进制-BCD优先编码器的引脚图和逻辑符号(HPRI表示最高值输入优先)

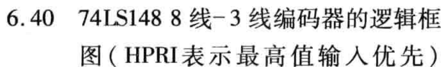


74LS148 8 线-3 线编码器

74LS148 是一个优先编码器，它有 8 个低电平有效输入和 3 个二进制低电平有效输出，如图 6.40 所示。这种芯片可以用于把八进制输入（回顾从 0 ~ 7 的八进制数）转换为 3 位二进制数。为了使芯片工作，*EI*（使能输入）必须为低电平。还有用于扩展功能的 *EO*（使能输出）和 *GS* 输出。当 *EI* 为低电平且所有输入（0 ~ 7）都为无效时，*EO* 为低电平。当 *EI* 为低电平且任何一个输入为有效输入时，*GS* 为低电平。

如图 6.41 所示，通过把高位编码器的 *EO* 连接到低位编码器的 *EI*，把相对应的输出连接到非-或门，就可以把 74LS148 扩展成一个 16 线-4 线编码器。*EO* 用做第 4 位最高有效位。这种特定的电路产生的结果是 4 位二进制高电平有效输出。





6.40　74LS148 8 线-3 线编码器的逻辑框图（HPRI表示最高值输入优先）

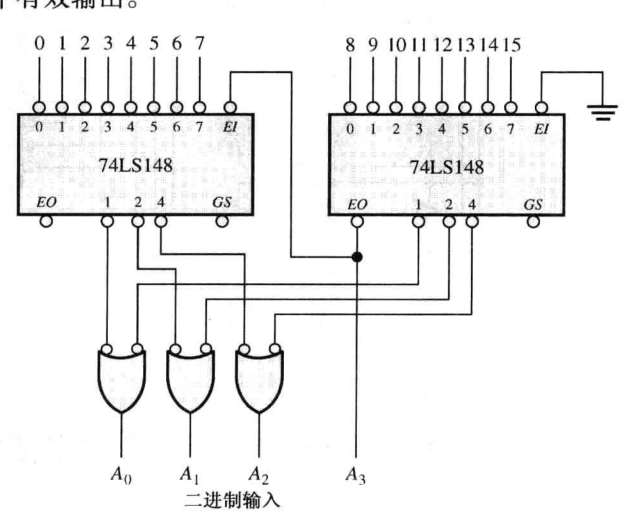


图 6.41　使用 74LS148 和外部逻辑电路的16线-4线编码器

例 6.11　如图 6.39 所示，如果 74HC147 中引脚 1、4 和 13 为低电平，试指出这 4 个输出的状态。所有其他输入为高电平。

解： 引脚 4 为十进制数输入的最高位，它为低电平，代表十进制数 7。因此，输出电平表示的是十进制数 7 的 BCD 码，其中 $\bar{A}_0$ 为最低有效位，$\bar{A}_3$ 为最高有效位。输出 $\bar{A}_0$ 为低电平，$\bar{A}_1$ 为低电平，$\bar{A}_2$ 为低电平，$\bar{A}_3$ 为高电平。

相关问题： 如果所有的输入都为低电平，那么 74HC147 的输出是什么？如果所有的输入都为高电平，那么其输出是什么？

6.6.2　应用举例

键盘编码器是一个典型的应用例子。例如，计算机键盘上的 10 个十进制数必须编码以便让逻辑电路处理。当按下其中一个键时，十进制数会被编码成相应的 BCD 码。如图 6.42所示，使用一片 74HC147 优先编码器，组成一个简单的键盘编码器。键盘上的按键为 10 个按钮开关，每个按钮都有一个上拉电阻（pull-up resistor）接电压 + *V*。当按键未被按下时，这些上拉电阻可以

确保线路为高电平。当按键被按下时，线路与地相连接，低电平加在相应的编码器输入上。按键0没有连接编码器输入，因为没有其他任何按键被按下时，BCD输出表示的是0。

编码器的BCD反码输出进入一个存储器，每一个连续的BCD码都会被存储起来，直到全部数字输入完毕。在以后的章节里，还会介绍存储BCD码和二进制数据的方法。

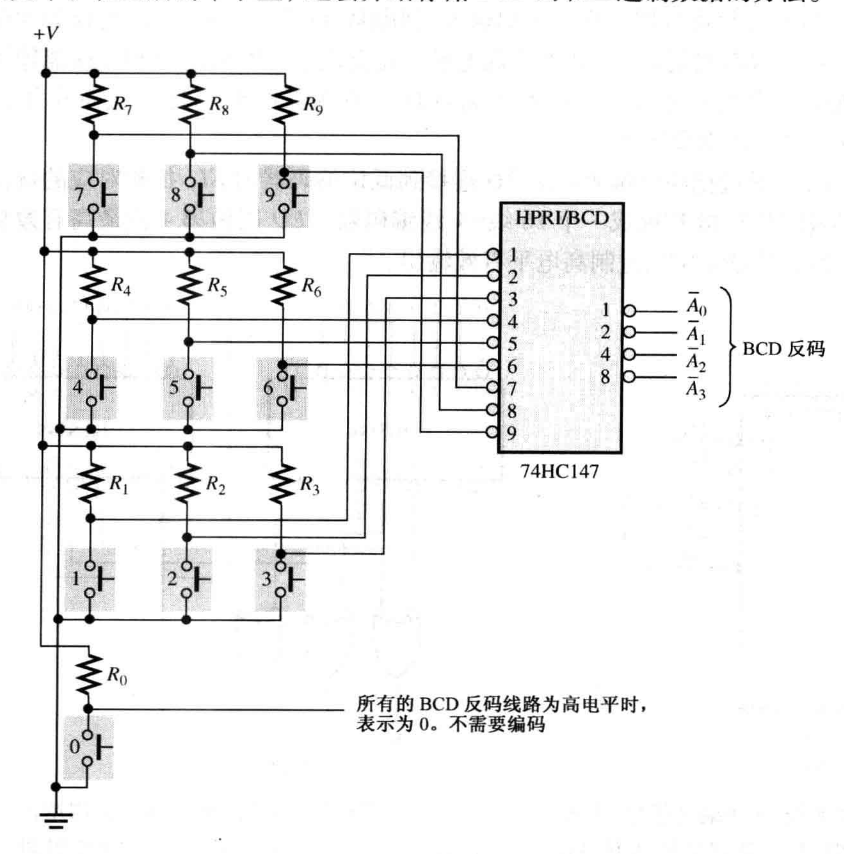


图6.42 一个简单的键盘编码器

6.6节 温故而知新

1. 假设一个高电平加在图6.38电路的第2个输入和第9个输入上，
   (a)输出线的状态是什么？
   (b)一个有效的BCD码表示什么？
   (c)在图6.38中，译码逻辑的限制是什么？
2. (a)当低电平加在图6.39的74HC147的引脚1和引脚5上时，输出是什么？
   (b)这个输出表示什么？

6.7 代码转换器

6.7.1 BCD－二进制转换

BCD－二进制转换的一种方法就是使用加法电路。这种基本转换的过程如下：

1. 使用一个二进制数来表示 BCD 码中每个位的数值或权。
2. 在 BCD 码中，所有用二进制数表示的位的权是 1 的相加。
3. 相加的结果就是与 BCD 码等效的二进制数，更简洁的运算描述为

BCD 位的权所表示的二进制数相加，相加的结果就是完整的二进制数。

可以对一个 8 位 BCD 码(表示两位十进制数)进行验证，以便于理解 BCD 码与二进制数之间的关系。例如，十进制数 87 用 BCD 码可以表示为

$$\underbrace{1000}_{8}\ \underbrace{0111}_{7}$$

最左边的 4 位组表示 80，最右边的 4 位组表示 7。也就是说，最左边的一组有一个 10 的权，最右边的一组有一个 1 的权。在每一组中，二进制数中每个位的权为

	十位数				个位数			
权：	80	40	20	10	8	4	2	1
位的名称：	B_3	B_2	B_1	B_0	A_3	A_2	A_1	A_0

每一个 BCD 位等效的二进制数就是整个 BCD 码中相应的位权所表示的一个二进制数。三种表示法由表 6.7 给出。

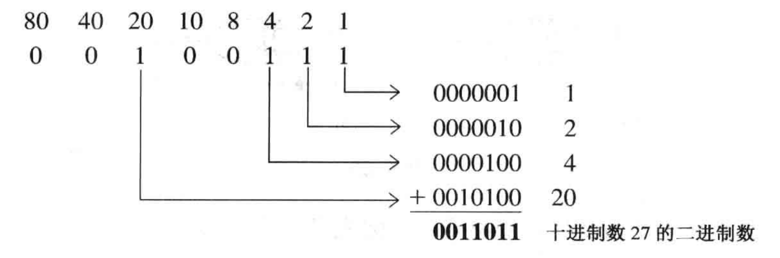
表 6.7　BCD 位权的二进制表示法

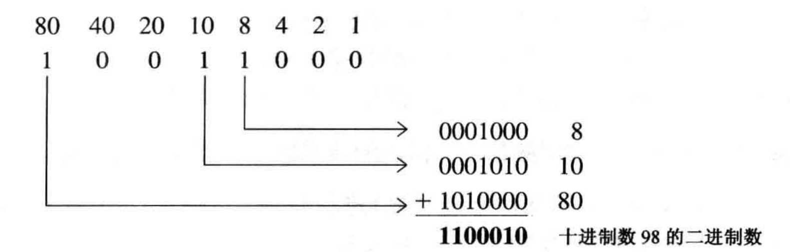
		最高有效位		二进制表示				最低有效位
BCD 位	BCD 权	64	32	16	8	4	2	1
A_0	1	0	0	0	0	0	0	1
A_1	2	0	0	0	0	0	1	0
A_2	4	0	0	0	0	1	0	0
A_3	8	0	0	0	1	0	0	0
B_0	10	0	0	0	1	0	1	0
B_1	20	0	0	1	0	1	0	0
B_2	40	0	1	0	1	0	0	0
B_3	80	1	0	1	0	0	0	0

如果在这个 BCD 码中，把所有为 1 的权所表示的二进制数相加，其结果就是与此 BCD 码相对应的二进制数。例 6.12 对此给出了解释。

例 6.12　把 BCD 码 00100111(十进制数为 27)和 10011000(十进制数为 98)转换为二进制数。

解：写出这个 BCD 码中出现的所有为 1 的权所表示的二进制数，然后把它们加起来。





相关问题：写出将 BCD 码 01000001 转换为二进制数的过程。

打开文件 EX06-12，运行仿真程序以观察 BCD-二进制逻辑电路的转换结果。

6.7.2 二进制-格雷码和格雷码-二进制转换

在第 2 章中，已经介绍过格雷码-二进制转换的基本过程。可以使用异或门来实现这种转换。其中的代码转换还可以用可编程逻辑器件编程实现。图 6.43 为一个 4 位二进制-格雷码转换器，图 6.44 为 4 位格雷码-二进制转换器。

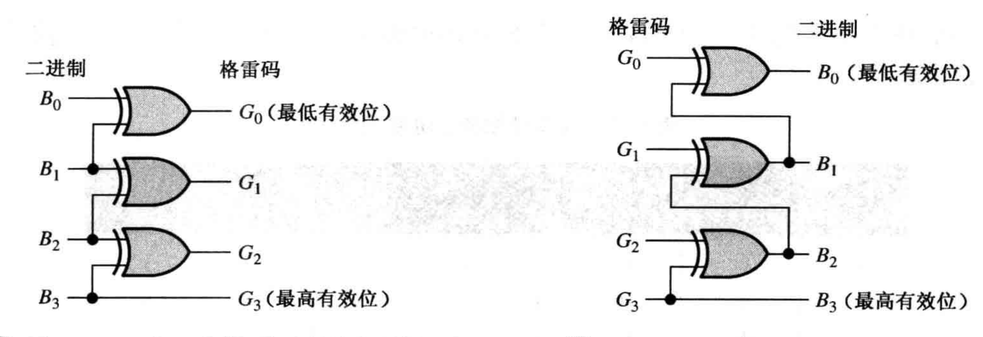


图 6.43 4 位二进制-格雷码转换逻辑。打开文件F06-43检验该操作

图 6.44 4 位格雷码-二进制转换逻辑。打开文件F06-44检验该操作

例 6.13 (a)使用异或门把二进制数 0101 转换成格雷码；

(b)使用异或门把格雷码 1011 转换成二进制数。

解：(a)如图 6.45(a)所示，0101_2的格雷码是 0111；

(b)如图 6.45(b)所示，格雷码 1011 的二进制数是 1101_2。

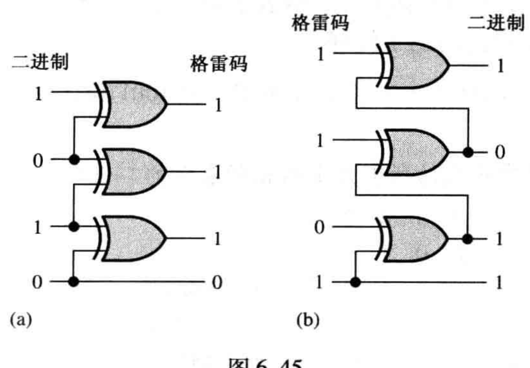


图 6.45

相关问题：把 8 位二进制数转换成格雷码，需要多少个异或门？

6.7 节　温故而知新

1. 把 BCD 码 10000101 转换为二进制数。
2. 画出 8 位二进制数转换为格雷码的逻辑框图。

6.8　多路复用器(数据选择器)

◇ 在一个多路复用器中，数据从几条线路传送到一条线路上。

图 6.46 为 4 位多路复用器的逻辑符号。注意，由于有两个数据选择位，所以有两条数据选择线，因此 4 条数据输入线中的任何一条都可被选中。

如图 6.46 所示，数据选择输入线(S)上的一个 2 位代码，允许被选中的数据输入线上的数据通过并传送到数据输出线上。如果二进制数 0($S_1=0$, $S_0=0$)加到数据选择线上，那么输入线上的数据 D_0 就会出现在输出数据线上。如果二进制数 1($S_1=0$, $S_0=1$)加到数据选择线上，那么输入线上的数据 D_1 就会出现在输出数据线上。如果二进制数 2($S_1=1$, $S_0=0$)加到数据选择线上，那么输入线上的数据 D_2 就会出现在输出数据线上。如果二进制数 3($S_1=1$, $S_0=1$)加到数据选择线上，那么输入线上的数据 D_3 就会和输出数据线接通。表 6.8 给出了这种运算的一个总结。

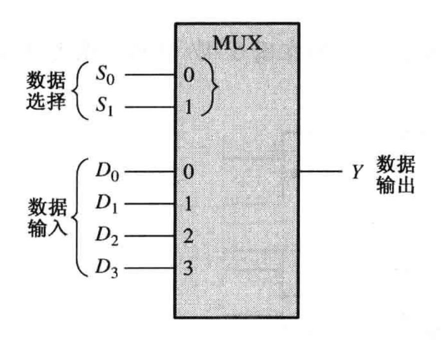


图 6.46　4 选 1 数据选择器/多路复用器的逻辑符号

表 6.8　4 选 1 多路复用器的数据选择

数据选择输入 S_1	S_0	选中的输入
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

现在，观察一下完成这种多路复用操作所需的逻辑电路。数据输出与被选中的数据输入是相同的。因此，可以得到一个关于数据输入和选择输入的输出逻辑表达式。

仅当 $S_1=0$ 和 $S_0=0$ 时，数据输出和 D_0 相等：$Y = D_0\overline{S}_1\overline{S}_0$。

仅当 $S_1=0$ 和 $S_0=1$ 时，数据输出和 D_1 相等：$Y = D_1\overline{S}_1S_0$。

仅当 $S_1=1$ 和 $S_0=0$ 时，数据输出和 D_2 相等：$Y = D_2S_1\overline{S}_0$。

仅当 $S_1=1$ 和 $S_0=1$ 时，数据输出和 D_3 相等：$Y = D_3S_1S_0$。

把这些项相或，数据输入的总的表达式为

$$Y = D_0\overline{S}_1\overline{S}_0 + D_1\overline{S}_1S_0 + D_2S_1\overline{S}_0 + D_3S_1S_0$$

这个等式的实现需要四个 3 输入与门、一个 4 输入或门和两个反相器，两个反相器用来产生 S_1 和 S_0 的反码，如图 6.47 所示。由于数据可以从输入线路中的任一条进行选择，所以也把这种电路看成是一个**数据选择器**。

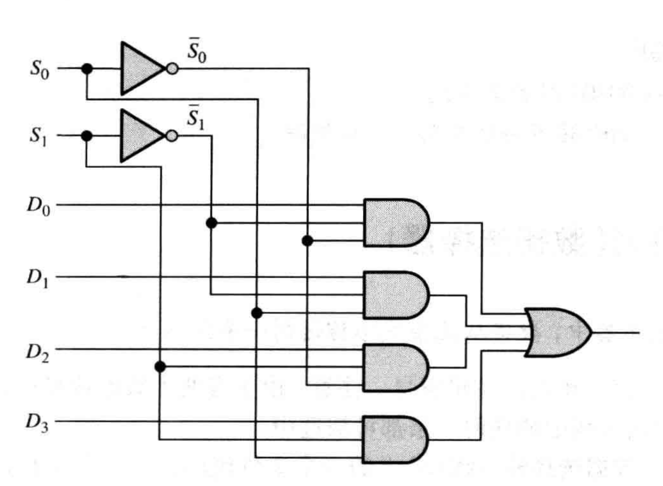


图 6.47 4 输入多路复用器的逻辑框图。打开文件 F06-47 检验该操作



计算机小知识

总线是一条计算机的多用途通路，在总线上计算机一端的电子信号传送到另一端。在计算机网络中，一条共享总线就是连接系统中所有的计算机且用于交换数据的通路。一条共享总线可以连接存储器和输入/输出设备，这样这些设备就可由系统中所有的计算机共享。对共享总线的访问是由总线判优器(类似于多路复用器)控制的，它每次只允许一台计算机共享系统的总线。

例 6.14 如图 6.48(a)所示的数据输入和数据选择波形，加在图 6.47 中的多路复用器上。请根据输入波形来确定输出波形。

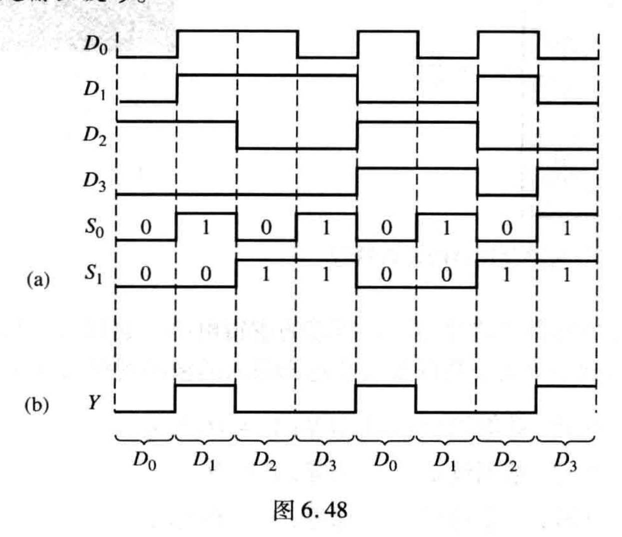


图 6.48

解：在每一个时间间隔期间，数据选择输入的二进制状态，确定了哪一条数据输入将被选中。注意，数据选择输入的二进制状态变化重复二进制序列 00，01，10，11，00，01，10，11，以此类推。所求的输出波形如图 6.48(b)所示。

相关问题：如果将图 6.48 中 S_0和 S_1的波形互换，请构建一个时序图，用来表示所有的输入和输出波形。



74HC157 四 2 输入数据选择器/多路复用器

74HC157 芯片，以及它的 LS 型号，都含有 4 个 2 输入多路复用器。这 4 个多路复用器的每一个都共享一个公共数据选择线和一个公共使能(*Enable*)端。由于每一个多路复用器只有两个输入被选择，所以单个数据选择输入已经足够。

$\overline{Enable}$ 输入端的一个低电平使得被选中的输入数据通过芯片传到输出端。而 $\overline{Enable}$ 输入端的一个高电平则阻止数据传到输出端；也就是说，多路复用器被禁止工作。

ANSI/IEEE 逻辑符号　74HC157 的引脚图如图 6.49(a)所示。74HC157 的 ANSI/IEEE 逻辑符号如图 6.49(b)所示。注意，这里的 4 个多路复用器被轮廓分割线分开，这 4 个多路复用器的公共输入由顶上有凹痕的框标出，称之为公共控制框。在 MUX(多路复用器)框上部的标注适用于下面其他的框。

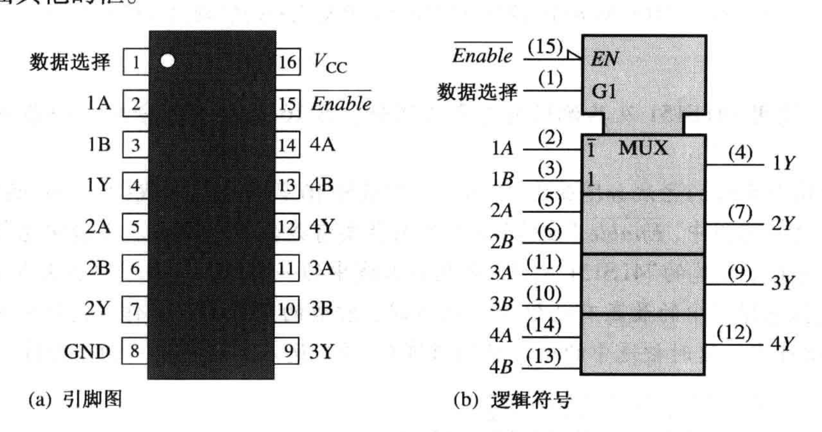


图 6.49　74HC157 四 2 输入数据选择器/多路复用器的引脚图和逻辑符号

注意在 MUX 框中的标注 1 和 $\overline{1}$ 及公共控制框中的 G1。这些标注是 ANSI/IEEE 标准 91-1984 确定的关联标注系统的一个例子。在这种情况下，G1 是表示数据选择输入和标注为 1 或 $\overline{1}$ 的数据输入的与运算关系。($\overline{1}$ 的意思是与的关系应用于 G1 输入的反码。)换句话说，当数据选择输入端为高电平时，多路复用器的 *B* 输入线被选中；当数据选择输入端为低电平时，多路复用器的 *A* 输入线被选中。“G”永远用来表示与关联。在本书适当的地方，还会介绍有关关联表示的其他方面。



74LS151 8 输入数据选择器/多路复用器

74LS151 有 8 个数据输入($D_0 \sim D_7$)，因此有 3 条数据选择或地址输入线($S_0 \sim S_2$)。为了选择 8 个数据输入($2^3=8$)中的任何一个，需要三个位。$\overline{Enable}$ 输入的低电平使得被选中的数据输入传送到输出端。注意，数据输出和它的反码都可得到。引脚图如图 6.50(a)所示，ANSI/IEEE 的逻辑符号如图 6.50(b)所示。在这种情况下，逻辑符号中不需要一个公共控制框，因为受控制的仅仅只有一个多路复用器，而不像 74HC157 那样有 4 个。逻辑符号中的标注 G $\frac{0}{7}$ 表示的是数据选择输入和从 0 ~ 7 的每一个数据输入相与的运算关系。

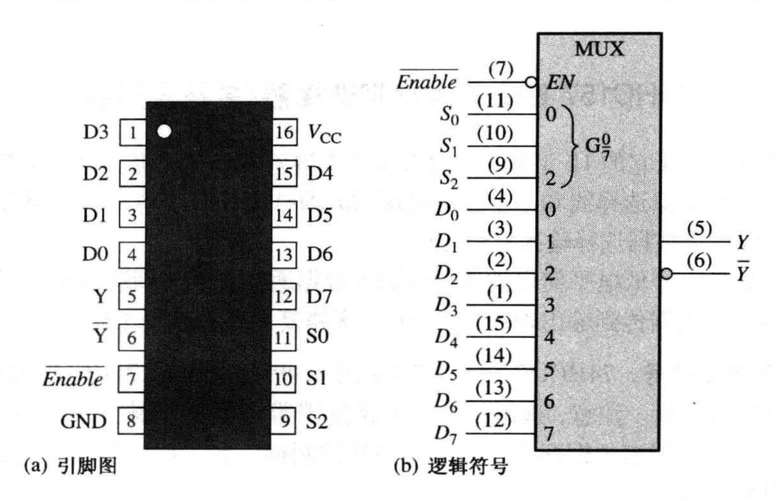


图 6.50　74LS151 8 输入数据选择器/多路复用器的引脚图和逻辑符号

例 6.15　使用 74LS151 及其他任何需要的逻辑，把 16 条数据线分配到一条单一的数据输出线上。

解：实现这个系统的方法如图 6.51 所示。为了选择 16 条数据输入线($2^4=16$)的其中一条，需要 4 个位。在这个应用中，$\overline{Enable}$(使能)输入用做最高有效数据选择位。当数据选择码中的最高有效位为低电平时，左边的 74LS151 工作，数据输入线中的一条($D_0 \sim D_7$)被另外 3 个数据选择位所选中。当数据选择码中的最高有效位为高电平时，右边的 74LS151 工作，数据输入线中的一条($D_8 \sim D_{15}$)被选中。这时被选中的输入数据通过非-或门进入到这条单一的输出线。

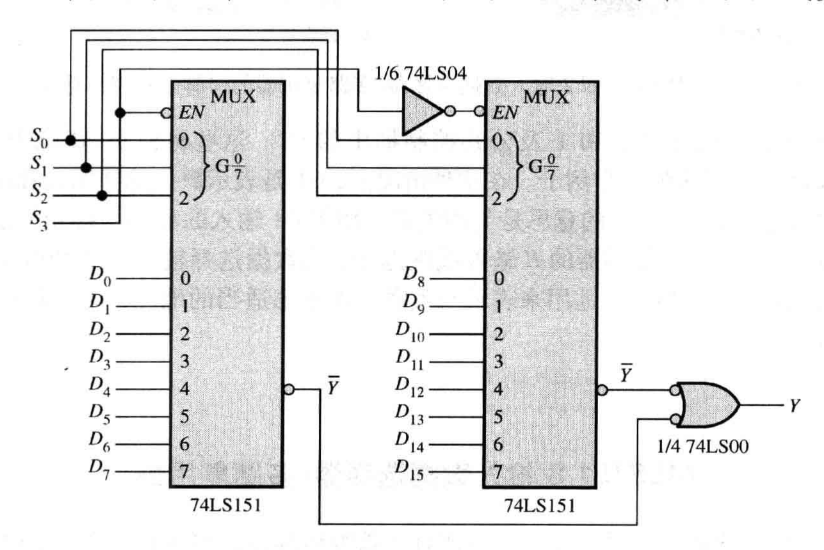


图 6.51　一个 16 输入多路复用器

相关问题：当需要选择如下的数据输入 D_0、D_4、D_8和 D_{13}时，确定选择输入的代码。

6.8.1　应用举例

7 段显示多路复用器　图 6.52 为把 BCD 数字分配到一个 7 段显示器上的一个简单方法。

在这个例子中，通过使用一个单一的 BCD－7 段译码器，把二位数字显示在 7 段显示器上。这个基本的显示分配方法可以扩展到显示任何位数。

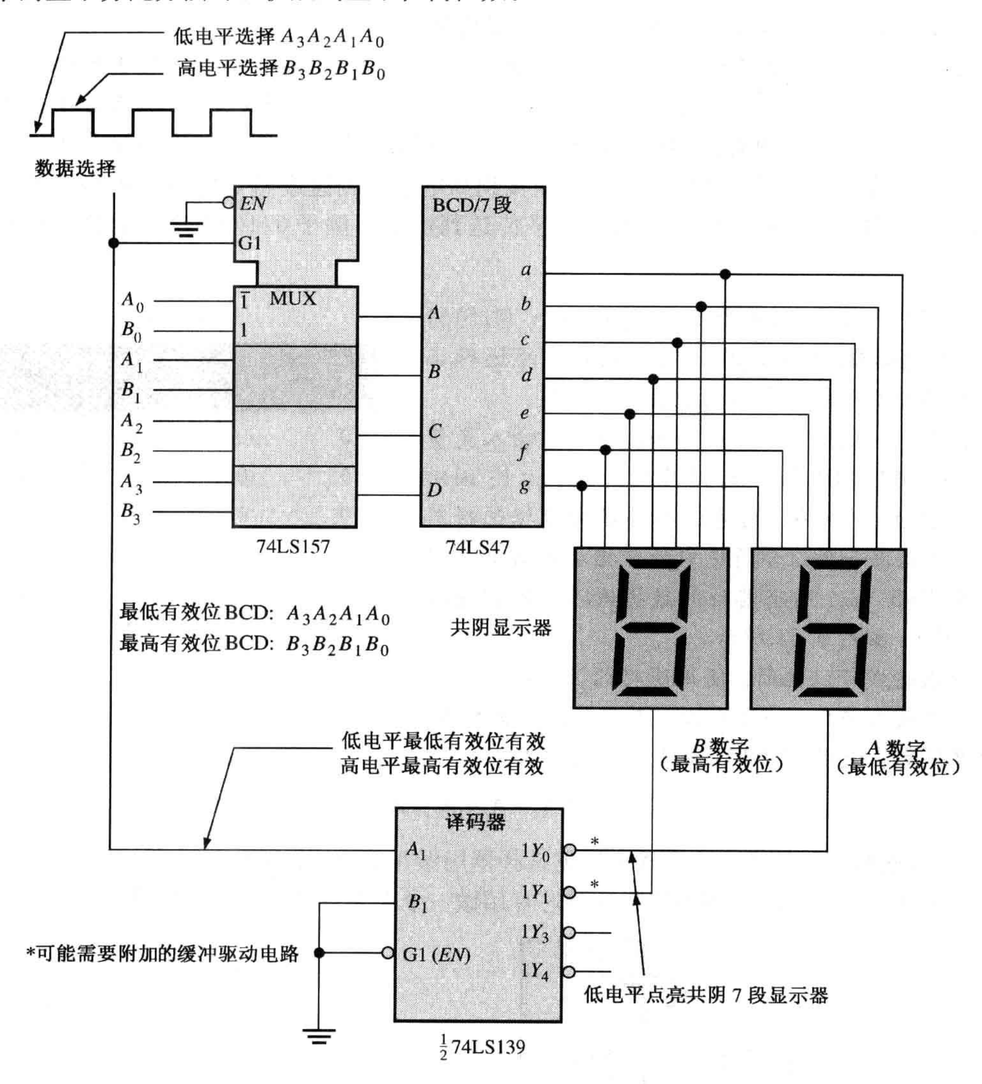


图 6.52　简单的 7 段显示器多路复用逻辑

基本处理过程如下：两个 BCD 数字（$A_3A_2A_1A_0$和 $B_3B_2B_1B_0$）加到多路复用器的输入。一个方波信号加在数据选择线上，当数据选择线为低电平时，A 数字的位（$A_3A_2A_1A_0$）上的信号传送到 74LS47 BCD－7 段译码器的输入。数据选择线上的低电平也使得 74LS139 2 线-4 线译码器的 A_1输入为低电平，这样译码器的 $1Y_0$输出有效，使得 A 数字在显示器上显示，显示器实际上是共阴连接。这时 A 数字点亮，B 数字熄灭。

当数据选择线为高电平时，B 位（$B_3B_2B_1B_0$）上的信号传送到 BCD-7 段译码器的输入。同样，由于 74LS139 译码器的 $1Y_1$输出有效，使得 B 数字在显示器上显示，这时 B 数字点亮，A 数字熄灭。重复的周期为数据选择端方波信号的频率。这个频率必须足够高（大约 30 Hz），以避免因数据的分配而产生视觉上的闪烁。

逻辑函数发生器　数据选择器/多路复用器的一种有用的应用就是乘积项之和形式的组合逻辑函数发生器。当以这种方式使用时，它可以代替一些分立的逻辑门，可以大大减少集成电路(IC)的数量，并且可以使得电路设计变得更加简单。

为了给予解释，可以用一个74LS151 8输入数据选择器/多路复用器来实现任何具体的3变量逻辑函数，这时3个变量连接到数据选择输入端，每一个数据输入端连接所需要的电平，这些电平由真值表中对应的输出函数所确定。例如，当3个变量的组合是$\overline{A}_2A_1\overline{A}_0$、输出函数为1时，数据输入端2(由010选中)连接到高电平。当这个特定的变量组合出现在数据选择输入线上时，数据输入端2上的高电平传送到输出。例子6.16将对阐明这个应用有所帮助。

例6.16　如表6.9所示，使用一个74LS151 8输入数据选择器/多路复用器来实现具体的逻辑函数。并和用分立逻辑门实现的方法进行比较。

解：注意，从这个真值表可知，如下的输入变量组合：001、011、101和110，对应的输出Y为1；而对于其他的组合，Y为0。因为这个函数由数据选择器来实现，所以每一个组合所选中的数据输入端必须连接到高电平(5 V)。所有其他的数据输入端则需连接低电平(地)，如图6.53所示。

除非表达式可以化简，使用逻辑门实现这个函数需要4个3输入与门、一个4输入或门和3个反相器。

表6.9

输入			输出
A_2	A_1	A_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

相关问题：使用74LS151实现下列表达式：

$$Y=\overline{A}_2\overline{A}_1\overline{A}_0+A_2\overline{A}_1\overline{A}_0+\overline{A}_2A_1\overline{A}_0$$

例6.16解释了如何把一个8输入数据选择器用做3变量逻辑函数发生器。实际上，使用连接数据输入的一个位(A_0)，还可以将这种芯片用做一个4变量逻辑函数发生器。

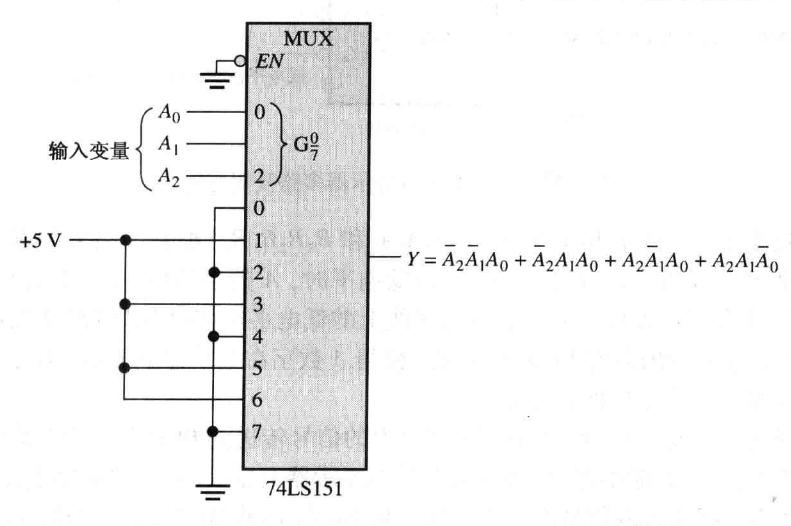


图6.53　数据选择器/多路复用器连接组成一个3变量逻辑函数发生器

一个4变量真值表有16个输入变量组合。当使用一个8位数据选择器时，每一个输入都会

被选择两次：第一次 A_0 为0，第二次 A_0 为1。把这个结论记下，可以运用下列规则（Y 为输出，A_0 为最低有效位）：

1. 如果某个输入变量的组合 $A_3A_2A_1$，在 $Y=0$ 时两次都选中了一个给定的数据输入端，那么把此数据输入端连接到地(0)。
2. 如果某个输入变量的组合 $A_3A_2A_1$，在 $Y=1$ 时两次都选中了一个给定的数据输入端，那么把此数据输入端连接到电压 $+V(1)$。
3. 如果某个输入变量的组合 $A_3A_2A_1$，在 $Y=0$ 和 $Y=1$ 时两次都选中了一个给定的数据输入端，并且如果 $Y=A_0$，那么把此数据输入端连接到 A_0。
4. 如果某个输入变量的组合 $A_3A_2A_1$，在 $Y=0$ 和 $Y=1$ 时两次都选中了一个给定的数据输入端，并且如果 $Y=\overline{A}_0$，那么把此数据输入端连接到 $\overline{A}_0$。

例6.17　使用一个74LS151 8输入数据选择器/多路复用器实现表6.10中的逻辑函数。并和用分立逻辑门实现的方法进行比较。

表6.10

十进制数	输入				输出
	A_3	A_2	A_1	A_0	Y
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

解：数据选择输入为 $A_3A_2A_1$。在表6.10的第1行中，$A_3A_2A_1=000$，$Y=A_0$。在第2行中，还是 $A_3A_2A_1=000$，$Y=A_0$。因此，A_0 与输入端0相连。在第3行中，$A_3A_2A_1=001$，此时 $Y=\overline{A}_0$。同样，在第4行中，$A_3A_2A_1=001$，$Y=\overline{A}_0$。因此，A_0 被反相并与输入端1相连。根据特定的规则，这种分析会一直持续下去直到与每个输入得到恰当的连接，如图6.54所示。

如果使用逻辑门实现，函数需要多达10个4输入与门、一个10输入或门和4个反相器。当然，可能的化简可以减少一些逻辑门。

相关问题：在表6.10中，如果输入都为0时，$Y=0$，并将表格中剩余行中 Y 的1和0交替排列，使用74LS151来实现这个逻辑函数。

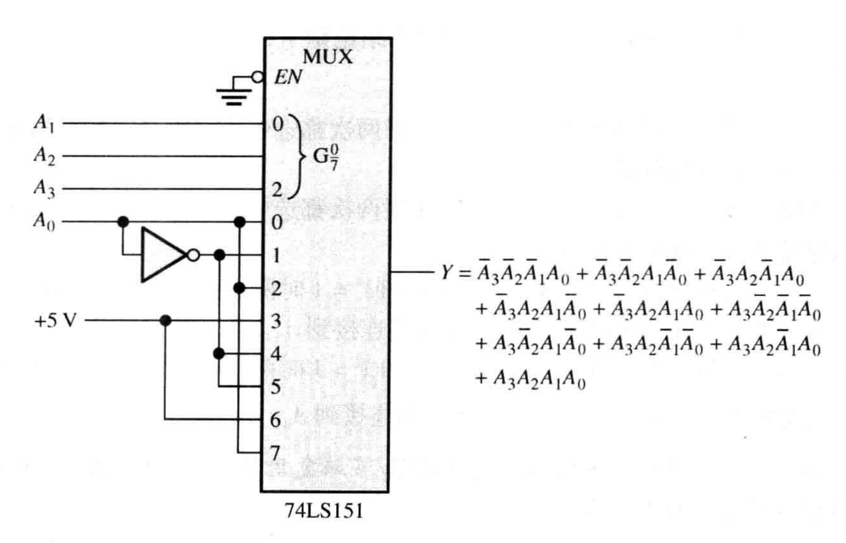


图6.54 数据选择器/多路复用器相连组成一个4变量逻辑函数发生器

6.8 节 温故而知新

1. 在图6.47中，$D_0 = 1$，$D_1 = 0$，$D_2 = 1$，$D_3 = 0$，$S_0 = 1$，$S_1 = 0$。输出是什么？
2. 判断芯片型号对应的逻辑器件。
   (a)74LS157　(b)74LS151
3. 74LS151的数据输入端 D_0 为交替的低电平和高电平，开始时 $D_0 = 0$。数据线以1 kHz的频率按二进制的序列变化(000, 001, 010, 以此类推)。使能端的输入为低电平。描述数据输出的波形。
4. 简单地描述图6.52中每个芯片的用途。
   (a)74LS157　(b)74LS47　(c)74LS139

6.9 多路分配器

◇ 在一个多路分配器中，数据从一条线路中被分配到几条线路上。

1线-4线多路分配器(DEMUX)的电路如图6.55所示。数据输入线与所有的与门连接。每一次，两条数据选择线只开通一个逻辑门，然后数据输入线上的数据通过这个选中的门，传送到相应的数据输出线上。

例6.18 串行数据输入波形(数据输入)和数据选择输入(S_0和S_1)如图6.56所示。试根据图6.55所示的这个多路分配器，确定从 D_0 到 D_3 的数据输出波形。

解： 输出波形如图6.56所示。注意，数据选择线通过一个二进制序列，使得每一个连续的输入位按照 D_0、D_1、D_2和 D_3的顺序变化。

相关问题： 如果时序图 S_0 和 S_1 波形反相，绘制这个多路复用器的时序图。

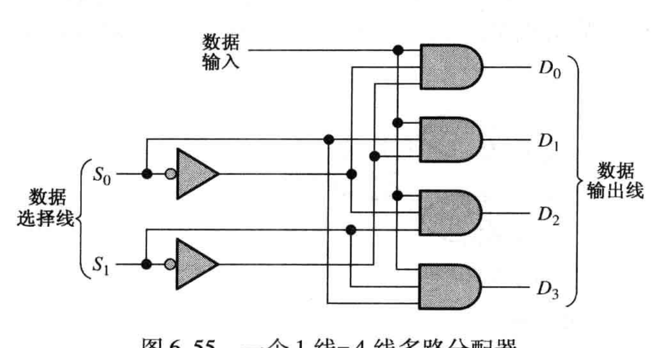


图 6.55 一个 1 线-4 线多路分配器

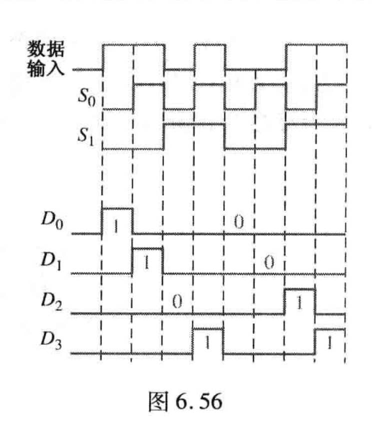


图 6.56



74HC154 多路分配器

前面介绍了 74HC154 译码器作为一个 4 线-16 线解码器的应用(见 6.5 节)。这种器件及其他的译码器，也可以用做多路分配器。当用做多路分配器时的逻辑符号如图 6.57 所示。在多路分配器的应用中，输入线用做数据选择线。其中一个片选输入用做数据输入线，其他片选输入为低电平时，使得图底部的内部非-与门工作。

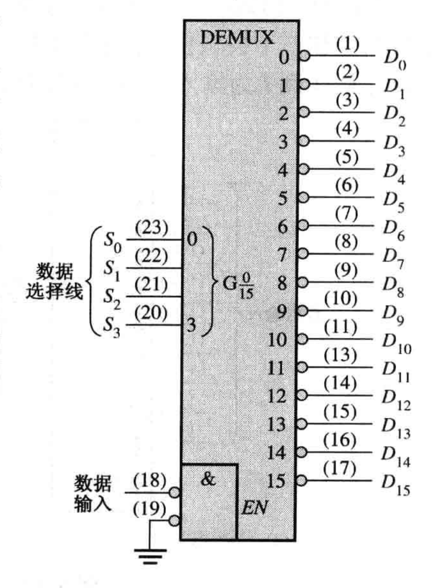


图 6.57 74HC154 译码器用做多路分配器

6.9 节 温故而知新

1. 一般情况下，多少个编码器可以组成一个多路分配器?
2. 图 6.57 中的多路分配器的数据线上有一个二进制数 1010，数据输入线处于低电平。输出线的状态是什么?

6.10 奇偶发生器/校验器

在第 2 章已经讲过，把一个奇偶校验位放于一组信号位中，使所有的 1 加起来为偶数或奇数(根据系统的情况而定)，这就是错误检测的奇偶校验方法。除了校验位，几种特殊的码还提供了固有误差的检测。

6.10.1 基本奇偶逻辑

◇ 用做错误检测的奇偶校验位指出在一个代码中，1 的个数是偶数还是奇数。

为了对一个给定的代码进行检测或者产生合适的奇偶校验，可以使用下列所给出的基本原理：

偶数个 1 相加(丢弃进位)的结果始终是 0，奇数个 1 相加的结果始终是 1。

因此，要确定一个给定的代码是偶校验还是奇校验，只需把代码中所有的位相加。如图 6.58(a)

所示，一个异或门可以产生两个位的模 2 和；如图 6.58(b)所示，3 个异或门相连，可以产生 4 个位的模 2 和；以此类推。当输入中 1 的个数为偶数时，输出 X 为 0(低电平)。当输入中 1 的个数为奇数时，输出 X 为 1(高电平)。

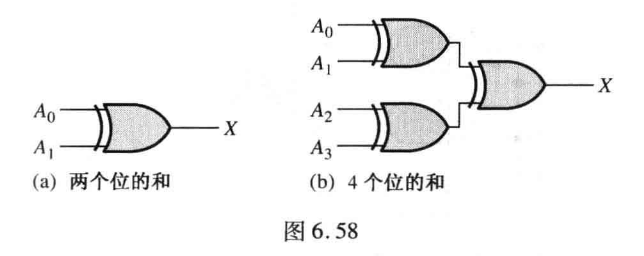
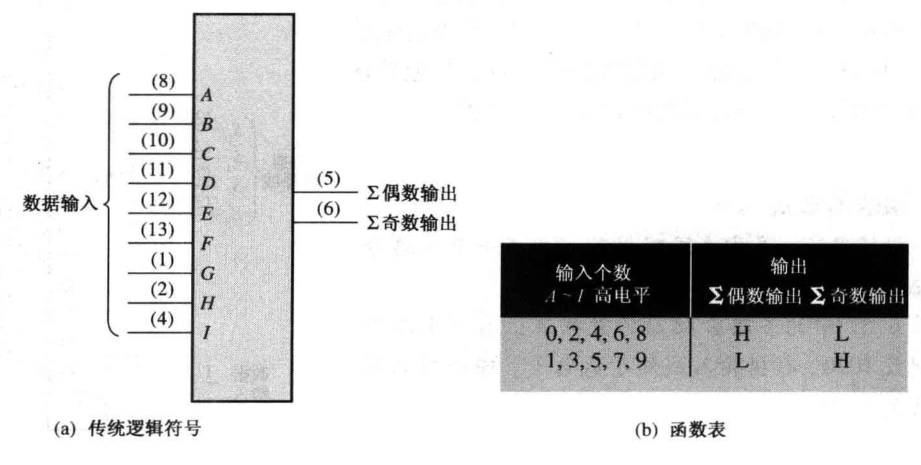


图 6.58



74LS280 9 位奇偶发生器/校验器

74LS280 的逻辑符号和函数表如图 6.59 所示。这种特殊的芯片可以用来对一个 9 位代码(8 个数据位和一个校验位)进行奇校验或偶校验，或者还可以对一个 9 位二进制代码产生一个校验位。从 A 到 I 为输入；当输入端有偶数个 1 时，Σ 偶数输出为高电平，Σ 奇数输出为低电平。



输入个数 A～I 高电平	输出 Σ偶数输出	Σ奇数输出
0, 2, 4, 6, 8	H	L
1, 3, 5, 7, 9	L	H

(b) 函数表

图 6.59 74LS280 9 位奇偶发生器/校验器

奇偶校验器 当这种芯片被用做一个偶校验器时，输入位的个数应该始终为偶数；当出现奇偶错误时，Σ 偶数输出为低电平，Σ 奇数输出为高电平。当这种芯片被用做一个奇校验器时，输入位的个数应该始终为奇数；当出现奇偶错误时，Σ 奇数输出为低电平，Σ 偶数输出为高电平。

奇偶发生器 如果把这种芯片用做一个偶发生器，奇偶校验位在 Σ 奇数输出获得，因为如果输入位的个数为偶数，则这个输出为 0；反之，如果输入位的个数为奇数，则这个输出为 1。当把这个器件用做一个奇发生器时，奇偶校验位在 Σ 偶数输出获取，因为当输入位的个数为奇数时，Σ 偶数输出为 0。

6.10.2 数据传输系统的错误检测

一个简单的数据传输系统如图 6.60 所示，用于阐明奇偶发生器/校验器的应用，以及多路复用器和多路分配器的应用，还可以解释某些应用中所需要的数据存储。

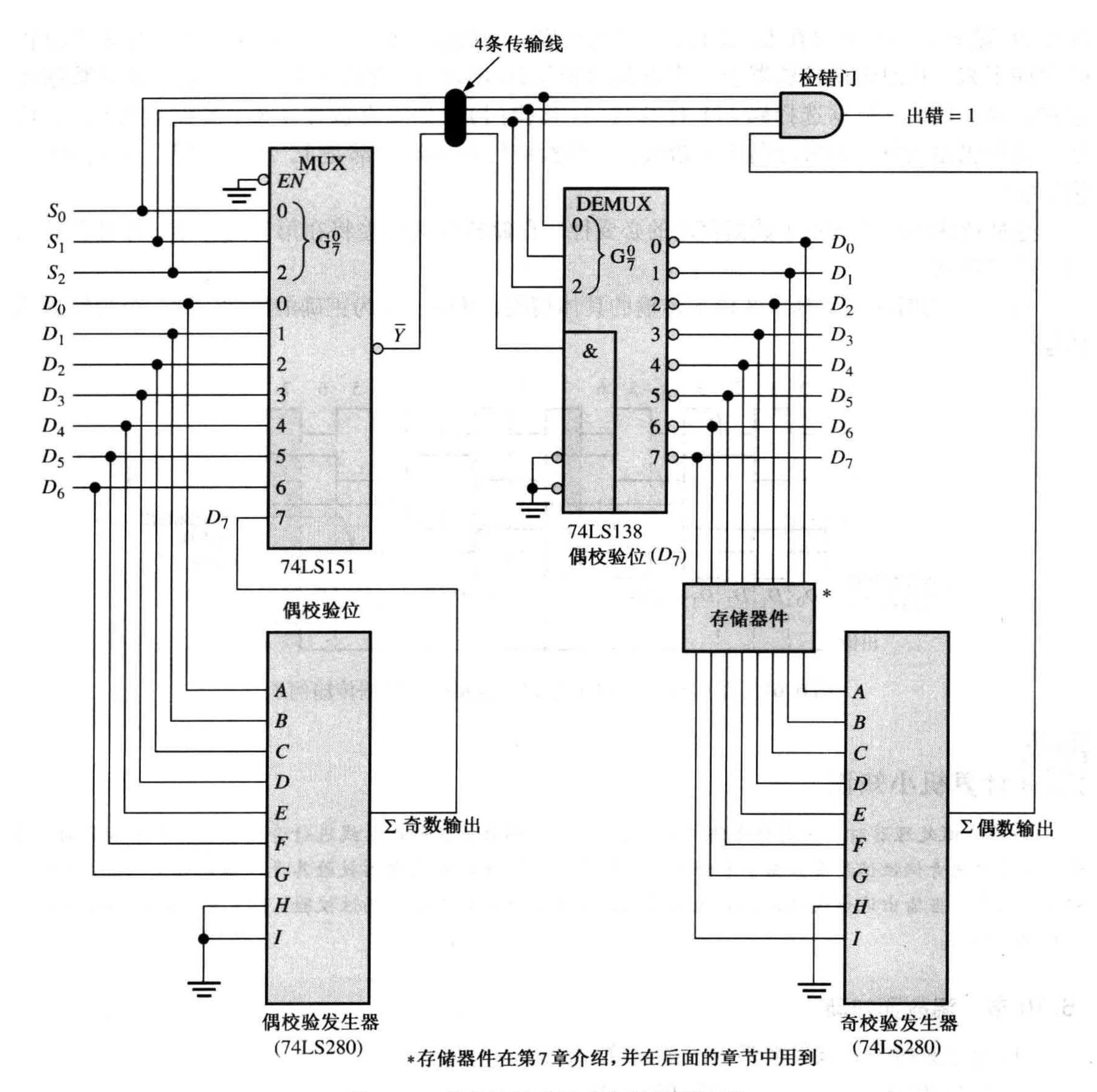


图 6.60　简单数据传输系统的错误检测

在此应用中，7 个数据源中的数字数据被多路复用器合并到一条单一的线路上，用做远距离传输。7 个数据位（从 D_0 ~ D_6）加到多路复用器的数据输入，与此同时加到偶发生器的输入端。奇偶发生器的 Σ 奇数输出用做偶校验位。如果输入 A 到 I 的 1 的个数为偶数，则这个偶校验位为 0；如果输入 A 到 I 的 1 的个数为奇数，则这个偶校验位为 1，这个位就是被传输码的 D_7。

数据选择输入在一个二进制序列里循环重复，从 D_0开始，每一个位以串行方式传输并到达传输线 $\bar{Y}$。在这个例子中，传输线由 4 条线路组成：1 条携带串行数据，3 条携带时序信号（数据选择）。当然，还有一些更复杂的传输时序信号的方法，可是为了解释基本原理，本章使用这个直接的方法。

在系统结尾部分的多路分配器上，数据选择信号和串行数据流加在了这个多路分配器上。由多路分配器把数据位按照与数据选择器相同的次序分配到输出线上。也就是说，D_0 出

现在 D_0 输出上，D_1 出现在 D_1 输出上，以此类推。奇偶校验位出现在 D_7 输出上。这 8 个位暂时存储下来，并加到偶校验器上。当奇偶检验位 D_7 出现时，奇偶校验器才有输入并且数据才会被存储。同时，数据选择码 111 打开与门(检错门)。如果奇偶性正确，Σ 偶数输出 0，检错门的输出也为 0。如果奇偶性是错误的，则检错门输入端全部是 1，结果检错门的输出(出错)为 1。

这种特殊的应用演示了数据存储的必要性。存储器件的用途将在第 7 章介绍，并且在后面的章节使用到。

图 6.61 的时序图解释了 8 位字传输的具体情况，其中一个为正确的传输，另一个为错误的传输。

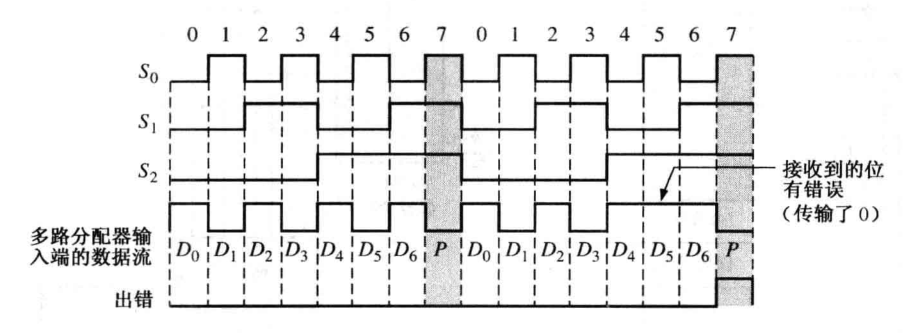


图 6.61　图 6.60 中系统有错误和没错误的数据传输例子



计算机小知识

Pentium 微处理器对内部进行奇偶校验，同样也对外部数据和地址总线进行奇偶校验。在读操作时，外部系统可以把奇偶性信息及数据字节合并在一起传送。Pentium 微处理器校验其奇偶性是否是偶数，并输出相应的信号。当输出地址代码时，Pentium 微处理器不会对地址进行奇偶性校验，但是它会对这个地址产生一个偶校验位。

6.10 节　温故而知新

1. 为下面的每个数字码添加一个偶校验位：
   (a)110100　　(b)01100011
2. 为下面的每个数字码添加一个奇校验位：
   (a)1010101　　(b)1000001
3. 检查每个偶校验码是否正确：
   (a)100010101　　(b)1110111001



数字系统应用

在这个数字系统应用中，我们将开始为一条繁忙的主干道和一条不太繁忙的边道设计一个交通信号灯的控制逻辑，给出这个系统需要的条件和总体框图。同时引入了一个状态图，用于确定操作的顺序。给出在应用中控制系统的组合逻辑部分。其余部分将在第 7 章和第 8 章给出。

时序上的要求

控制逻辑将控制一个交通信号灯的时序，交通信号灯为一条繁忙主干道和一条不太繁忙的边道的交叉口的信号灯。下面给出的是系统所需的时序。这些时序由图 6.62 解释。

- 主干道的绿灯至少亮 25 秒或只要边道没有车辆绿灯就亮。
- 边道的绿灯亮一直到边道没有车辆为止，最多亮 25 秒。
- 在主干道和边道的绿灯变为红灯的间隔时间里，警示黄灯亮 4 秒。

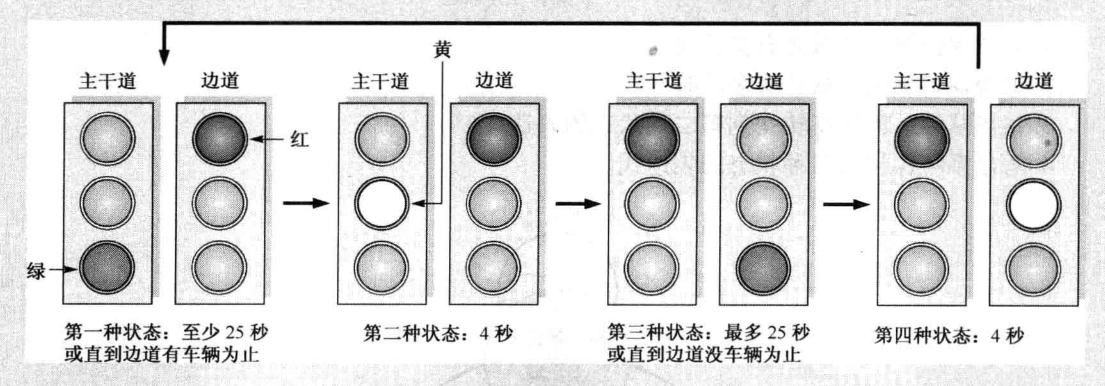


图 6.62 交通信号灯的时序图要求

状态图

根据状态要求设计出一个描述整个系统运行的状态图。通常一个状态图给出状态的次序、每个状态的情况，以及从一个状态到下一个状态变化的要求。虽然图 6.62 是一个基本形式的状态图，但是需要一个更为详细类型的状态图，用于描述具体状态变化的逻辑和布尔表达式。

变量的定义 确定系统状态如何按时序变化的变量定义如下：

- V_s 为边道上出现车辆。
- T_L 为 25 秒定时器(长定时器)启动。
- T_S 为 4 秒定时器(短定时器)启动。

使用变量的反码表示相反的条件。

状态描述 图 6.63 给出了一个状态图。图中 4 种状态的每一种都被赋予一个 2 位的格雷码。环形箭头表示系统保持状态，状态之间的箭头表示系统的当前状态变为下一个状态。与状态图中每个箭头相关联的布尔表达式或变量表示状态的变化情况，状态保持不变或从一种状态变化到下一种状态。

第一种状态 格雷码为 00。在这种状态下(S_1)，主干道上为绿灯亮，边道上为红灯亮。在长定时器启动或边道上没有车辆的条件下保持此状态 25 秒。这种情况由 $T_L + \bar{V}_s$ 表示。当长定时器关闭和边道上有车辆时，系统进入下一个状态，由 $\bar{T}_L V_s$ 表示。

第二种状态 格雷码为 01。在这种状态下(S_2)，主干道上为黄灯(警示灯)亮，边道上为红灯亮。当短定时器启动(T_S)时，系统保持在此状态 4 秒，这种状态由 T_S 表示。当短定时器关闭时，系统进入下一个状态。这这种状态由 $\bar{T}_S$ 表示。

第三种状态 格雷码为11。在这种状态下(S_3),主干道上为红灯亮,边道上为绿灯亮。在长定时器启动和边道上有车辆($T_L V_s$)时,系统保持这种状态25秒。这种状态由 $T_L V_s$ 表示。当25秒时间超过或边道上没有车辆通过时,系统进入下一个状态。这种状态由 $\overline{T}_L + \overline{V}_s$ 表示。

第四种状态 格雷码为10。在这种状态下(S_4),主干道上为红灯亮,边道上为黄灯亮。短定时器启动,系统保持此状态4秒。这种状态由 T_S 表示。当短定时器关闭时,系统回到第一种状态。这种状态由 $\overline{T}_S$ 表示。

1. 系统保持第一种状态有多长时间?
2. 系统保持第四种状态有多长时间?
3. 写出从第一种状态转变到第二种状态的表达式。
4. 写出系统保持第二种状态的表达式。

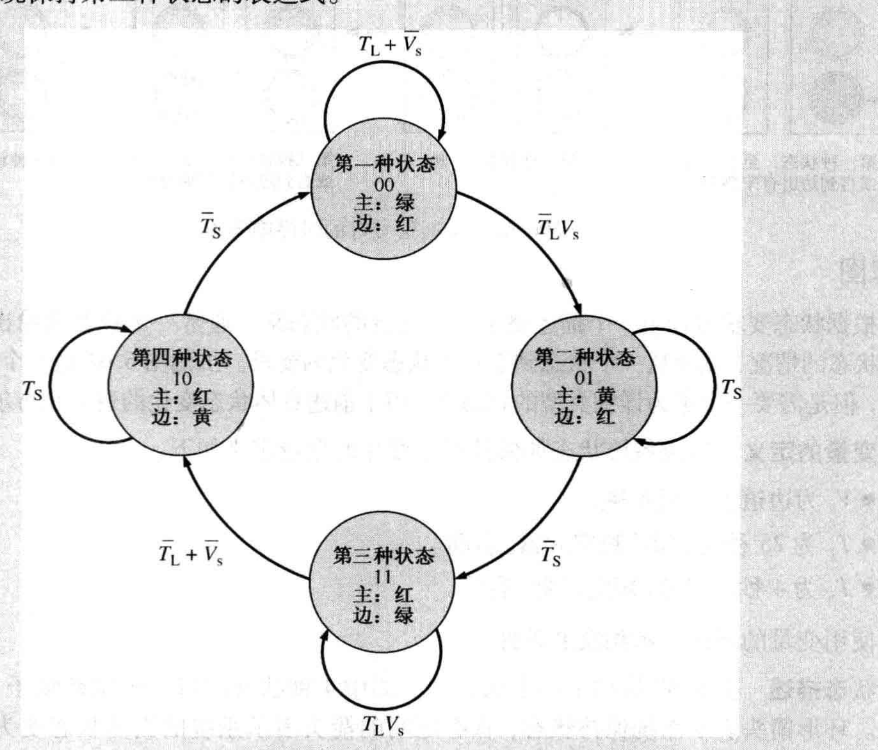


图6.63 给出格雷码顺序的交通信号灯控制系统的状态图

系统框图

交通信号灯控制系统由三部分组成:组合逻辑、时序逻辑和定时电路,如图6.64所示。

系统的组合逻辑部分提供打开和关闭信号灯的输出。也提供启动长、短定时器开始时的触发输出。系统逻辑的输入时序表示的4种状态由状态框图描述。系统的这部分汇集了本例的具体应用。

系统的定时电路部分提供25秒和4秒定时输出,也提供时钟输出以使时序逻辑通过此4种状态。时序逻辑产生表示4种状态的2位格雷码。

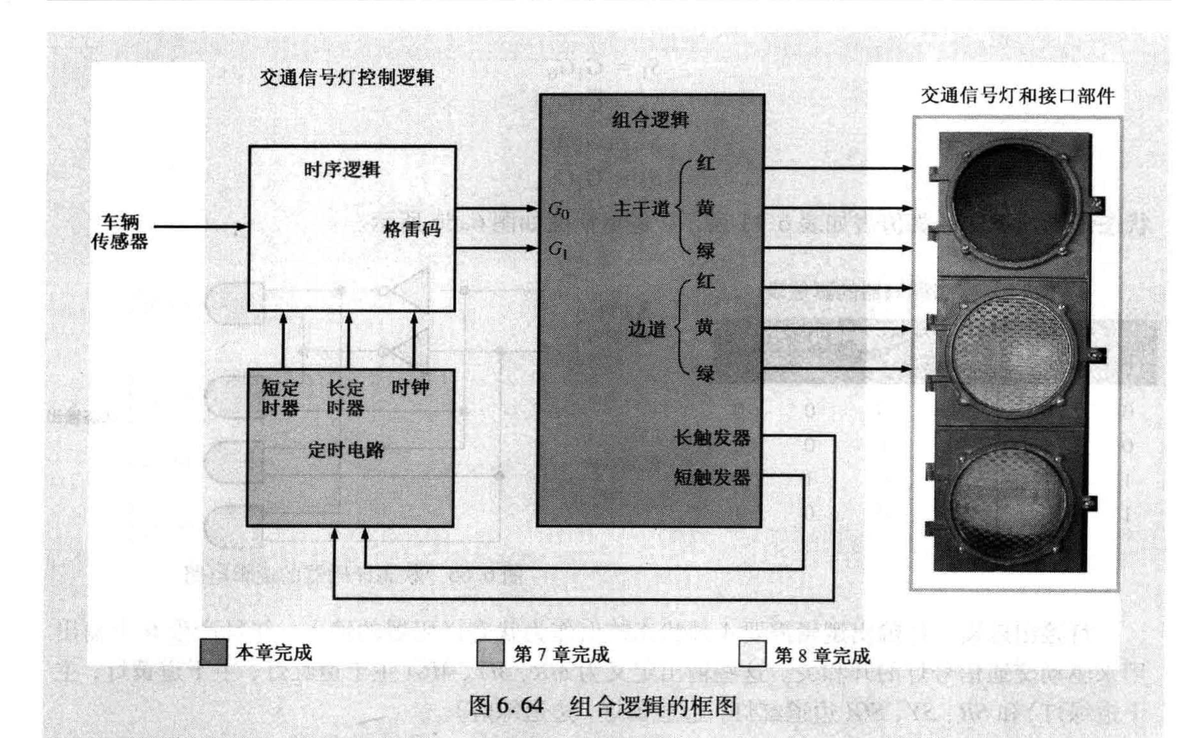


图 6.64　组合逻辑的框图

组合逻辑

如图 6.65 所示，组合逻辑电路由一个状态译码器、灯输出逻辑和触发逻辑组成。

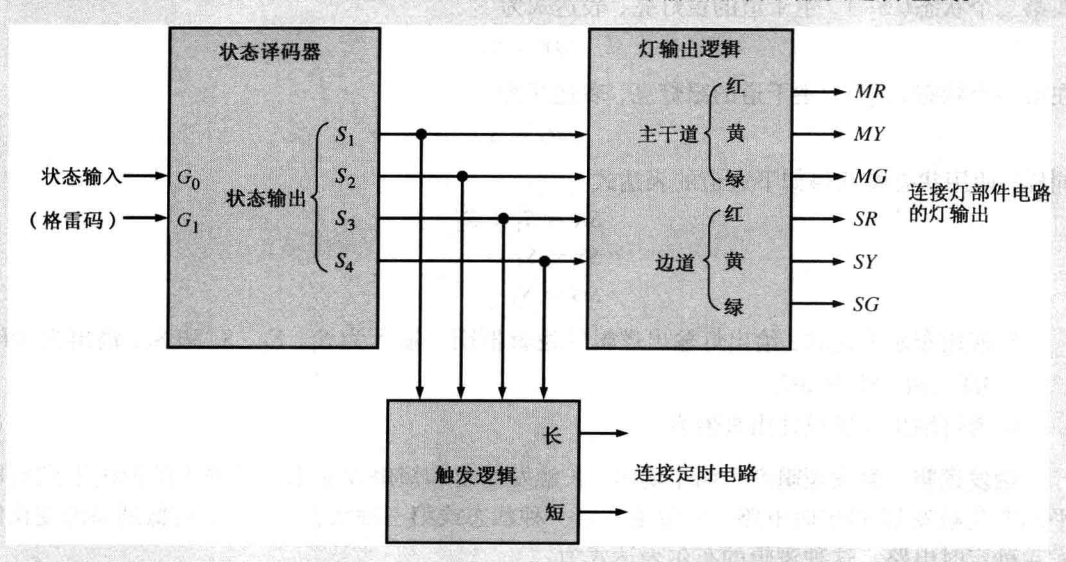


图 6.65　组合逻辑的框图

状态译码器　对时序逻辑中的 2 位格雷码进行译码，确定系统处在 4 种状态中的哪一种状态。状态译码器的输入为 2 位格雷码 G_1 和 G_0。有 4 种输出状态 S_1、S_2、S_3 和 S_4。对于 4 种输入的每一个，有且仅有一个对应的输出。对应于输入的输出表达式为

$$S_1 = \overline{G}_1\overline{G}_0$$
$$S_2 = \overline{G}_1 G_0$$
$$S_3 = G_1 G_0$$
$$S_4 = G_1\overline{G}_0$$

状态译码器逻辑的真值表如表 6.11 所示，逻辑框图如图 6.66 所示。

表 6.11 状态译码器的真值表

状态输入		状态输出			
G_1	G_0	S_1	S_2	S_3	S_4
0	0	1	0	0	0
0	1	0	1	0	0
1	1	0	0	1	0
1	0	0	0	0	1

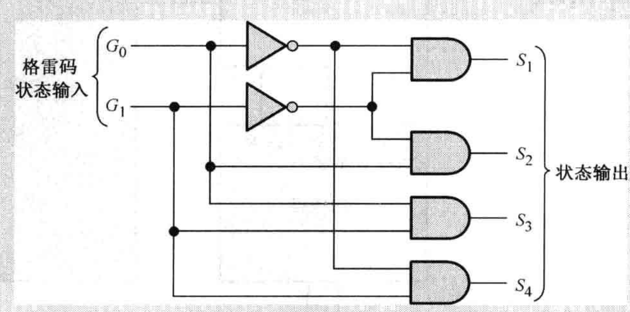


图 6.66 状态译码器的逻辑电路

灯输出逻辑 灯输出逻辑需要 4 种状态输出作为状态译码器的输入，并且产生 6 个输出用来驱动交通信号灯的开和关。这些输出定义为 *MR*、*MY*、*MG*(主干道红灯，主干道黄灯，主干道绿灯)和 *SR*、*SY*、*SG*(边道红灯，边道黄灯，边道绿灯)。

参见状态框图，在第三个状态(S_3)和第四个状态(S_4)，主干道的红灯亮，因此布尔表达式为

$$MR = S_3 + S_4$$

在第二个状态(S_2)，主干道的黄灯亮，表达式为

$$MY = S_2$$

在第一个状态(S_1)，主干道的绿灯亮，表达式为

$$MG = S_1$$

同样，使用状态图获得如下边道的表达式：

$$SR = S_1 + S_2$$
$$SY = S_4$$
$$SG = S_3$$

5. 使用布尔表达式，给出灯输出逻辑的逻辑框图。输入为 S_1、S_2、S_3 和 S_4，输出为 *MR*、*MY*、*SR*、*SY* 和 *SG*。
6. 为灯输出逻辑设计出真值表。

触发逻辑 触发逻辑产生两个输出：长触发输出和短触发输出。长输出的低电平到高电平的变化触发 25 秒定时电路。短输出在第一种状态或第三种状态时，电平的低到高的变化触发 4 秒定时电路。这种逻辑的布尔表达式为

长触发 $= S_1 + S_3$

短触发 $= S_2 + S_4$

7. 使用布尔表达式，给出触发逻辑的逻辑框图。
8. 为触发逻辑设计出真值表。
9. 通过组合状态译码器、灯输出逻辑和触发逻辑，给出完整的组合逻辑框图。

计算机仿真

图6.67给出了交通信号灯控制系统的完整组合逻辑部分的计算机仿真屏幕，它包括状态译码器、灯输出逻辑和触发逻辑。使用开关来仿真格雷码输入，并使用探针来指示组合逻辑的输出状态。开关和探针仅用于计算机仿真，不属于逻辑电路部分。

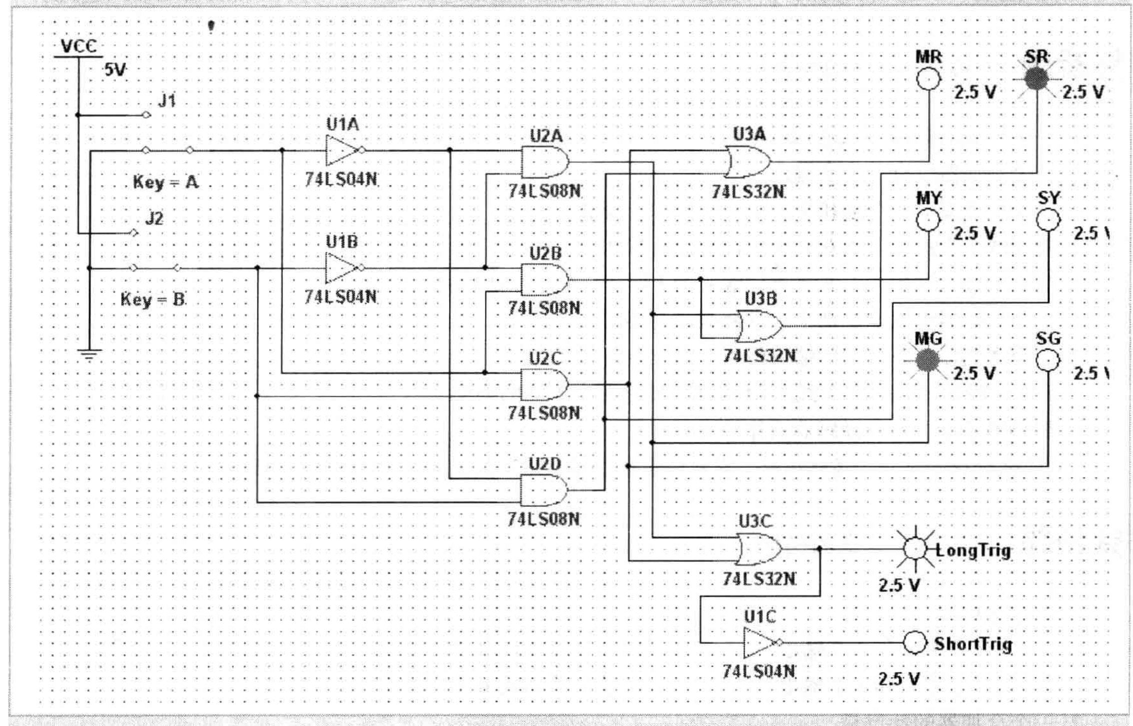


图6.67 第一种状态的组合逻辑的Multisim电路的屏幕界面

打开系统应用示例的Multisim文件SAA06。运行交通信号控制系统的组合逻辑部分的仿真，观察灯序列的4种状态的每一种的运行结果。

把知识用于实践

需要有一个人行道控制按钮，用于在主干道和边道打开黄色警示灯4秒，打开红灯15秒。(a)修改状态框图，用于附加的功能；(b)设计附加的逻辑电路。

关键词

级联 连接两块或多块芯片，用于扩展一个逻辑电路能力的方法。

译码器 把代码信息转换为熟悉的或非编码形式的一个数字电路。

多路分配器(DEMUX) 把数字数据以特定的时序从一个输入传送到不同输出中的一条输出线上的一种数字电路。

编码器 把信息转换为编码的形式的一种数字电路。

全加器 把两个数位和进位相加，产生一个和与输出进位的一种数字电路。

假信号 持续时间很短的一个电压或电流尖峰，通常是随意产生的和不希望出现的。

半加器 把两个数位相加，产生一个和与输出进位的一种数字电路。

超前进位 一种二进制加法的方法，各级加的进位预先进行运算，以此减少进位的传输延迟。

数据选择器(多路复用器，MUX) 把数字数据以特定的时序从多个输入中的一个传送到一条输出线上的一种数字电路。

校验位 附加在每组信息位的一个数位，使得每组信息位的1的个数为奇数或偶数。

异步进位 二进制加法的一种方法，每个加法器的输出进位是下一个高位加法器的输入进位。

判断题 （答案在本章的结尾。）

1. 一个半加器对两个二进制位相加。
2. 一个半加器仅有一个和输出。
3. 一个全加器对三个位相加，并且产生两个输出。
4. 使用两个全加器可以对两个4位数进行加法。
5. 当两个输入位都为1并且进位输入为1时，一个全加器的和输出为0。
6. 一个比较器可以确定两个二进制数是否相等。
7. 一个译码器检测特定的输入位组合的存在。
8. 4线-10线译码器和1线-10线译码器是两个不同的类型。
9. 编码器基本上实现译码器操作的逆过程。
10. 数据选择器是一个逻辑电路，它允许数字信息从单一的信号线传送到几条线路上。

自测题 （答案在本章的结尾。）

1. 一个半加器的特点是
   (a)有2个输入和2个输出　　(b)有3个输入和2个输出
   (c)有2个输入和3个输出　　(d)有2个输入和1个输出
2. 一个全加器的特点是
   (a)有2个输入和2个输出　　(b)有3个输入和2个输出
   (c)有2个输入和3个输出　　(d)有2个输入和1个输出
3. 一个全加器的输入为$A=1$、$B=1$、$C_{in}=0$，则其输出为
   (a)$\Sigma=1$，$C_{out}=1$　　(b)$\Sigma=1$，$C_{out}=0$
   (c)$\Sigma=0$，$C_{out}=1$　　(d)$\Sigma=0$，$C_{out}=0$
4. 一个4位并行加法器可以用于相加
   (a)两个4位二进制数　　(b)两个2位二进制数
   (c)一次4位　　(d)顺序相加4个位
5. 如果将一个4位并行加法器扩展为一个8位加法器，则必须
   (a)使用4个4位不相连的加法器。
   (b)使用两个4位加法器，并将其中一个加法器的和输出与另一个加法器的进位输入相连。
   (c)使用8个4位不相连的加法器。
   (d)使用两个4位加法器，并将其中一个加法器的进位输出与另一个加法器的进位输入相连。
6. 如果一个74LS85比较器的输入为$A=1011$、$B=1001$，则其输出为
   (a)$A>B=0$，$A<B=1$，$A=B=0$　　(b)$A>B=1$，$A<B=0$，$A=B=0$
   (c)$A>B=1$，$A<B=1$，$A=B=0$　　(d)$A>B=0$，$A<B=0$，$A=B=1$
7. 一个输出为低电平有效输出的16选1译码器，在它的十进制12输出出现了一个低电平，求这个译码器的输入是什么？

(a)$A_3A_2A_1A_0=1010$　　(b)$A_3A_2A_1A_0=1110$

(c)$A_3A_2A_1A_0=1100$　　(d)$A_3A_2A_1A_0=0100$

8. 一个 BCD－7 段解码器的输入为 0100，则其有效输出为

(a)a, c, f, g　　(b)b, c, f, g

(c)b, c, e, f　　(d)b, d, e, g

9. 如果一个八进制-二进制优先编码器的 0、2、5、6 输入都为有效电平，则其高电平有效的二进制输出为

(a)110　　(b)010　　(c)101　　(d)000

10. 在一般情况下，一个数据选择器有

(a)一个数据输入、几个数据输出和几个选择输入

(b)一个数据输入、一个数据输出和一个选择输入

(c)几个数据输入、几个数据输出和几个选择输入

(d)几个数据输入、一个数据输出和几个选择输入

11. 数据选择器与下列哪种器件基本相同？

(a)译码器　　(b)多路分配器　　(c)多路复用器　　(d)编码器

12. 下列代码中，哪一个代码含有偶校验？

(a)10011000　　(b)01111000　　(c)1111111

(d)11010101　　(e)以上所有选项　　(f)选项(b)和(c)

习题（奇数题的答案在本书的结尾。）

6.1 节　基本加法器

1. 如图 6.4 所示的全加器，如果这个全加器的输入为下列选项，请确定每个输出门的逻辑状态（1 或 0）：

(a)$A=1, B=1, C_{in}=1$

(b)$A=0, B=1, C_{in}=1$

(c)$A=0, B=1, C_{in}=0$

2. 如果下列选项为一个全加器所产生的输出，则其输入是

(a)$\Sigma=0, C_{out}=0$　　(b)$\Sigma=1, C_{out}=0$

(c)$\Sigma=1, C_{out}=1$　　(d)$\Sigma=0, C_{out}=1$

3. 如果一个全加器的输入为下列选项，请确定这个全加器的输出：

(a)$A=1, B=0, C_{in}=0$　　(b)$A=0, B=0, C_{in}=1$

(c)$A=0, B=1, C_{in}=1$　　(d)$A=1, B=1, C_{in}=1$

6.2 节　并行二进制加法器

4. 并行加法器如图 6.68 所示，请根据电路中的逻辑操作分析来确定它的和，使用两个输入数字的手算的方法来检验其结果。

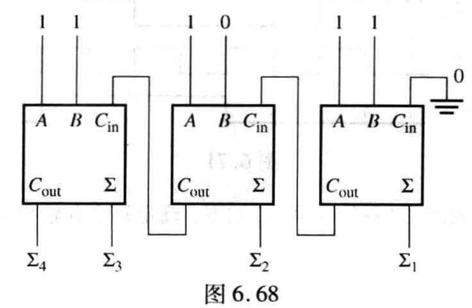


图 6.68

5. 根据图 6.69 所示的电路及输入，重复习题 4。

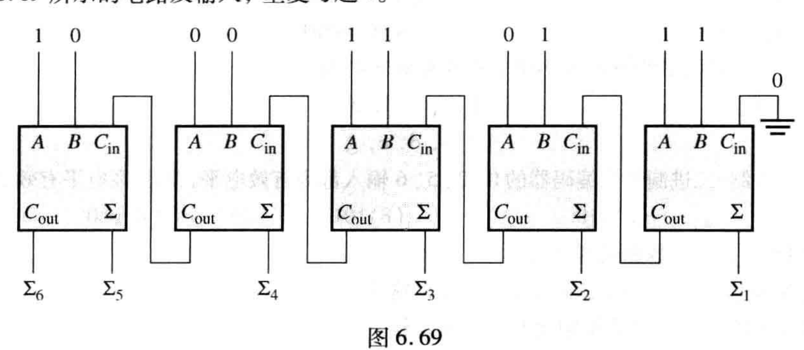


图 6.69

6. 如图 6.70 所示为一个 4 位电路，可以在计算机中进行加和减的运算(正数以原码的形式，负数以反码的形式)。

(a) 当 $\overline{Add}/Subt.$ 为高电平时，电路的功能是什么?

(b) 当 $\overline{Add}/Subt.$ 为低电平时，电路的功能是什么?

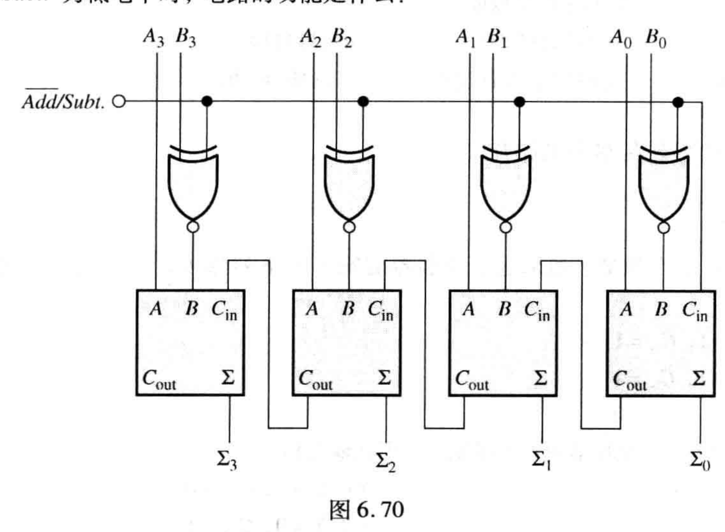


图 6.70

7. 对于图 6.70 的电路，假设输入 $\overline{Add}/Subt.$= 1、A = 1001 和 B = 1100，那么输出是什么?

8. 一个 2 位加法器的输入波形如图 6.71 所示。建立一个时序图，并根据这个时序图中的输入，确定和的波形及进位输出波形。

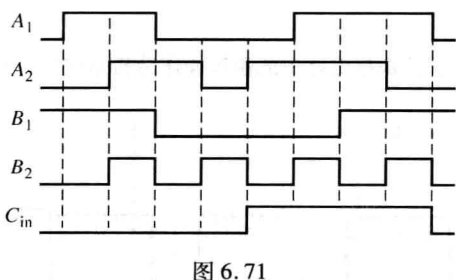


图 6.71

9. 4 位并行加法器输入位的顺序(最右位为第一)如下，确定每个和输出中的位顺序。

A_1	1001
A_2	1110
A_3	0000
A_4	1011
B_1	1111
B_2	1100
B_3	1010
B_4	0010

10. 在检测一个 74LS283 4 位并行加法器的过程中，其引脚的电压电平如下所示：
1 – 高，2 – 高，3 – 高，4 – 高，5 – 低，6 – 低，7 – 低，9 – 高，10 – 低，11 – 高，12 – 低，13 – 高，14 – 高 和 15 – 高。请确定这个集成电路是否正常工作？

6.3 节　异步进位与超前进位加法器

11. 在 8 位并行异步进位加法器中，8 个全加器的每一个有下面的传输延迟：
A 到 Σ 和 C_{out}：40 ns(纳秒)　　B 到 Σ 和 C_{out}：40 ns(纳秒)
C_{in} 到 Σ：35 ns(纳秒)　　C_{in} 到 C_{out}：25 ns(纳秒)
为两个 8 位数的加法确定其最大的全部运算时间。
12. 给出使得图 6.18 的 4 位超前加法器变成 5 位加法器的附加逻辑电路。

6.4 节　比较器

13. 这个比较器的波形如图 6.72 所示，请确定这个比较器的输出波形($A=B$)。

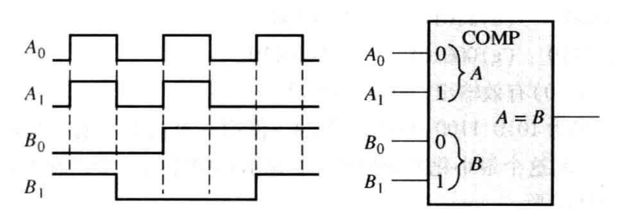


图 6.72

14. 一个 4 位比较器如图 6.73 所示，请根据所示输入，绘出每一个输出波形。其输出为高电平有效。

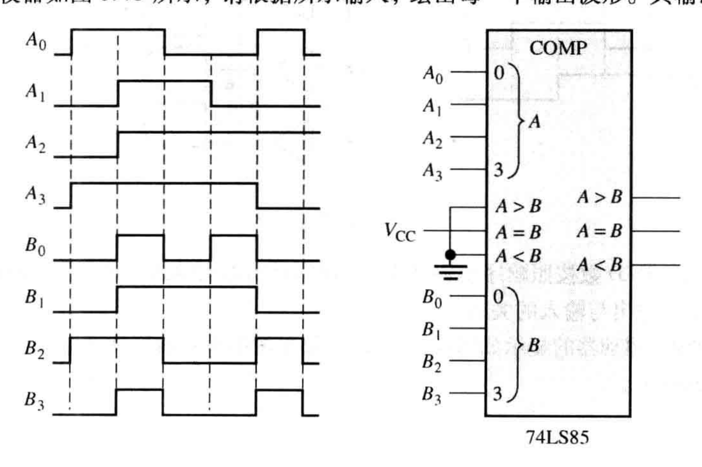


图 6.73

15. 如图 6.22 所示，请根据下列每一组二进制数，确定比较器的输出状态。
(a) $A_3A_2A_1A_0=1100$　$B_3B_2B_1B_0=1001$

(b)$A_3A_2A_1A_0=1000$ $B_3B_2B_1B_0=1011$

(c)$A_3A_2A_1A_0=0100$ $B_3B_2B_1B_0=0100$

6.5 节 译码器

16. 如图6.74所示，当译码门的每一个输出都为一个高电平时，试确定这个译码门输入的二进制数，其最高有效位为A_3。

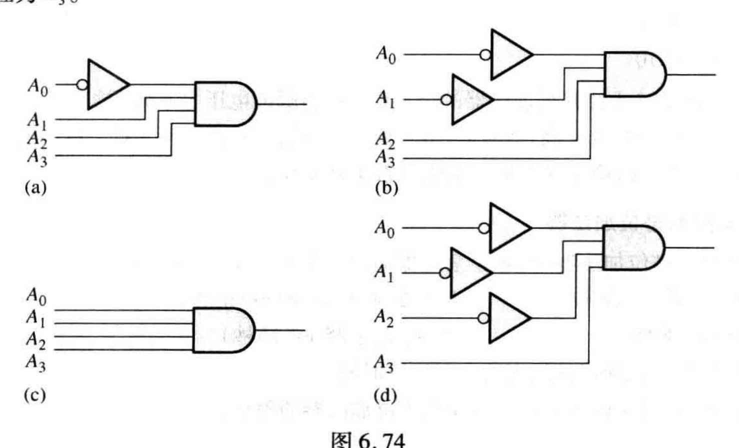


图6.74

17. 如果需要一个高电平(1)有效输出，试确定下列代码的译码门逻辑：

(a)1101 (b)1000 (c)11011 (d)11100

(e)101010 (f)111110 (g)000101 (h)1110110

18. 如果需要一个低电平(0)有效输出，重复习题17。

19. 只希望检测出的代码为1010、1100、0001和1011，使用一个高电平有效输出来表示它们的出现。使用一个单一输出，绘制这个最小化的译码逻辑，显示其中任何一个出现在输入的代码。如果为其他代码，输出必须为低电平。

20. 图6.75为加在译码逻辑电路的输入波形，请根据输入波形绘出其输出波形。

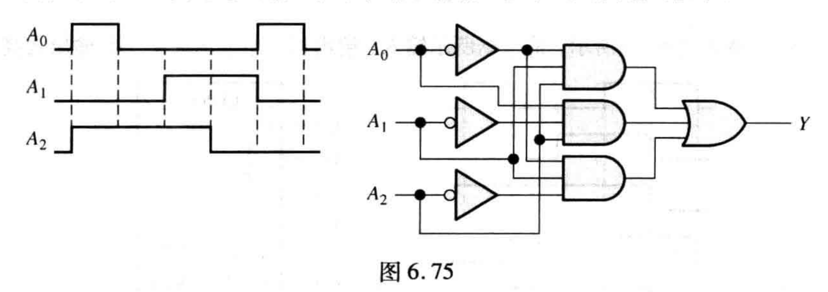


图6.75

21. 如图6.76所示，BCD数按照顺序加到BCD-十进制译码器的输入。请绘制一幅时序图，显示输出之间的关系及每个输出与输入的关系。

22. 一个7段译码器/驱动器的显示如图6.77所示。如果图中所示波形加在芯片的输入，请确定显示器显示的数字的顺序。

6.6 节 编码器

23. 十进制-BCD编码器逻辑如图6.38所示。假设输入9和输入3都为高电平。其输出代码是什么？是否为一个有效的BCD码(8421码)？

24. 一个74HC147编码器的引脚2、5和12都为低电平。如果其他的输入都为高电平，其输出产生的BCD码是什么？

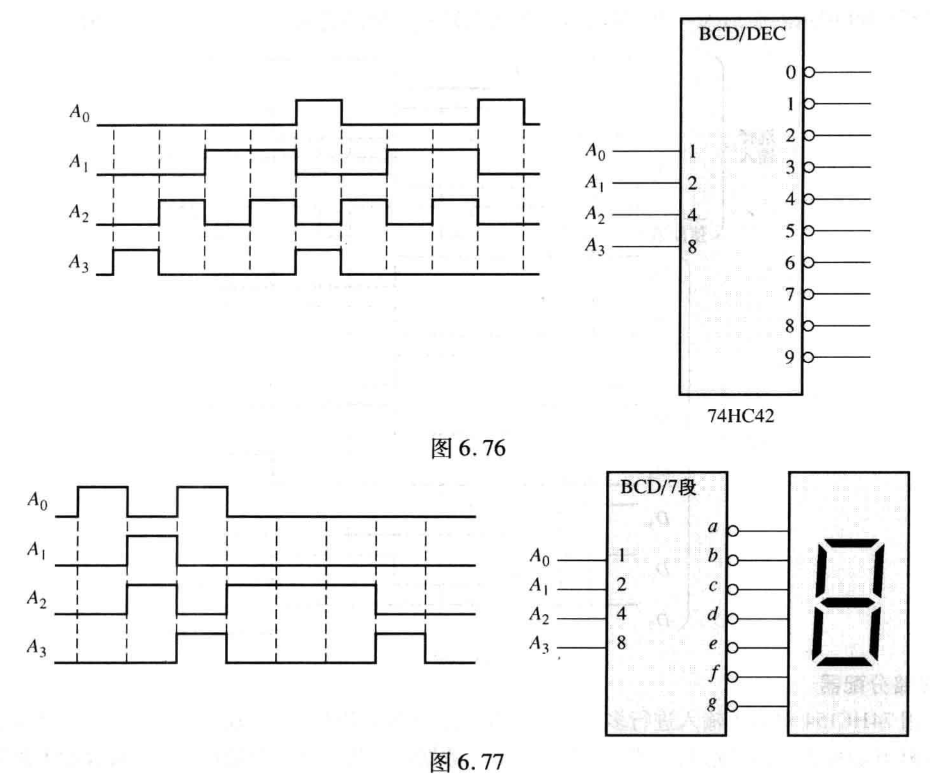


图 6.76

图 6.77

6.7 节 代码转换器

25. 将下列的十进制数转换成 BCD 和二进制数：

(a)2 (b)8 (c)13 (d)26 (e)33

26. 写出将一个 10 位二进制数转换成格雷码所需要的逻辑，并用这个逻辑将下列二进制数转换成格雷码：

(a)1010101010 (b)1111100000 (c)0000001110 (d)1111111111

27. 写出将一个 10 位格雷码转换成二进制数所需要的逻辑，并用这个逻辑将下列格雷码转换成二进制数：

(a)1010000000 (b)0011001100 (c)1111000111 (d)0000000001

6.8 节 多路复用器(数据选择器)

28. 如图 6.78 所示的多路复用器，请根据下列输入状态确定其输出：

$D_0=0$，$D_1=1$，$D_2=1$，$D_3=0$，$S_0=1$，$S_1=0$

29. 如果图 6.78 的数据选择输入为图 6.79 所示的波形，请根据习题 28 给出的数据输入确定其输出波形。

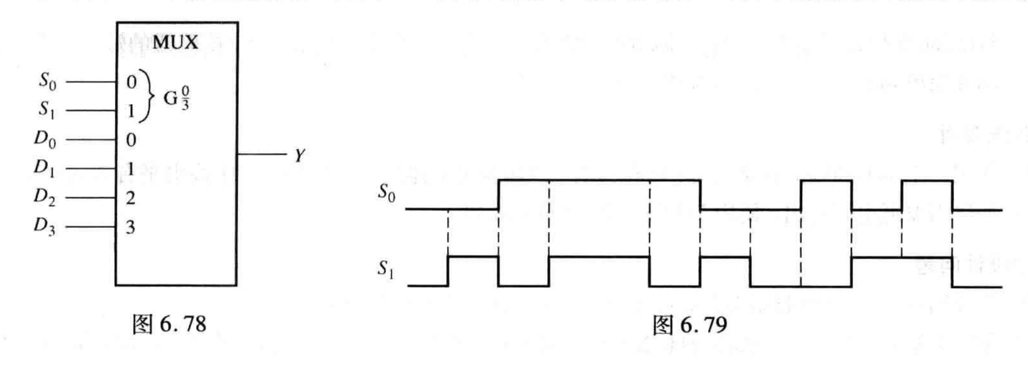


图 6.78

图 6.79

30. 如图 6.80 所示的波形为一个 74LS151 8 输入多路复用器的输入，试绘制 Y 输出波形。

图 6.80

6.9 节 多路分配器

31. 使用 74HC154 对如下输入进行多路分配的应用，绘制总的时序图(输入和输出)：数据选择输入从 0000 开始按连续的二进制计数顺序循环变化，数据输入为一个串行数据流，带有表示十进制数 2468 的 BCD 数据。最低有效位(8)在顺序的第一位，由于最低有效位处在第一位，所以它出现在 4 位输出的第一位。

6.10 奇偶发生器/校验器

32. 4 位奇偶校验逻辑的波形如图 6.81 所示。请根据其输入确定输出波形。当偶校验发生时，应有多少个位时间，如何得出这个位时间？这个时序图含有 8 个位时间。

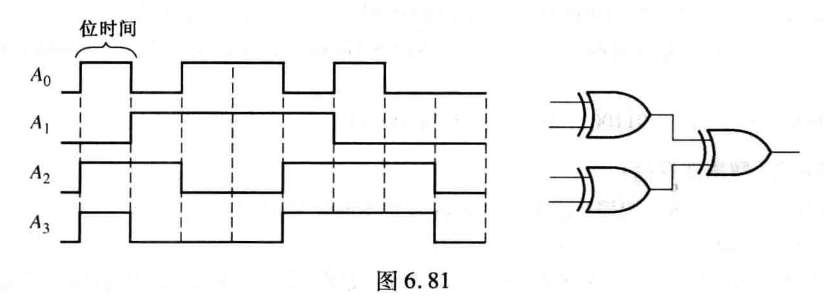


图 6.81

33. 74LS280 9 位奇偶发生器/校验器如图 6.82 所示。请根据这个奇偶发生器/校验器的输入，确定其 Σ 偶数输出和 Σ 奇数输出。参见图 6.59 中的真值表。

数字系统应用

34. 使用一个 74LS00 芯片(双与非门)和其他一些需要的电路为给出的输入产生高电平有效的输出。

35. 如果需要低电平输出，使用 74LS00 完成灯输出逻辑。

特殊的设计问题

36. 修改图 6.52 的 7 段显示多路系统的设计，完成两个附加位的设计。

37. 使用表 6.2，写出一个全加器的和 Σ 与进位输出 C_{out} 的乘积项之和表达式。使用卡诺图化简该表达式，

然后使用反相器和与-或逻辑实现它。给出如何使用 74LS151 数据选择器替代与-或逻辑的方法。

38. 使用 74LS151 数据选择器完成表 6.12 定义的逻辑功能。

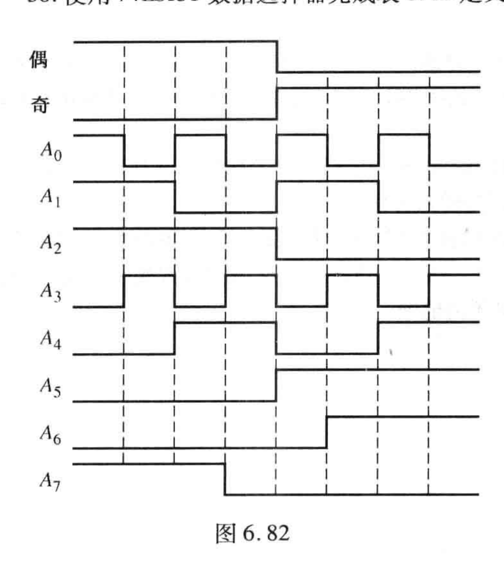


图 6.82

表 6.12

输入				输出
A_3	A_2	A_1	A_0	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

39. 使用两个图 6.14 的 6 个投票位置加法器，设计一个 12 个投票位置的系统。

40. 图 6.83 为药片灌装系统的加法器框图，执行从计数器得到的 8 位二进制数和从寄存器 B 得到的 16 位二进制数的加法。加法器的运算结果再送回到寄存器 B 中。使用 74LS283 完成这个功能，并且画出包括引脚编号的完整逻辑框图。

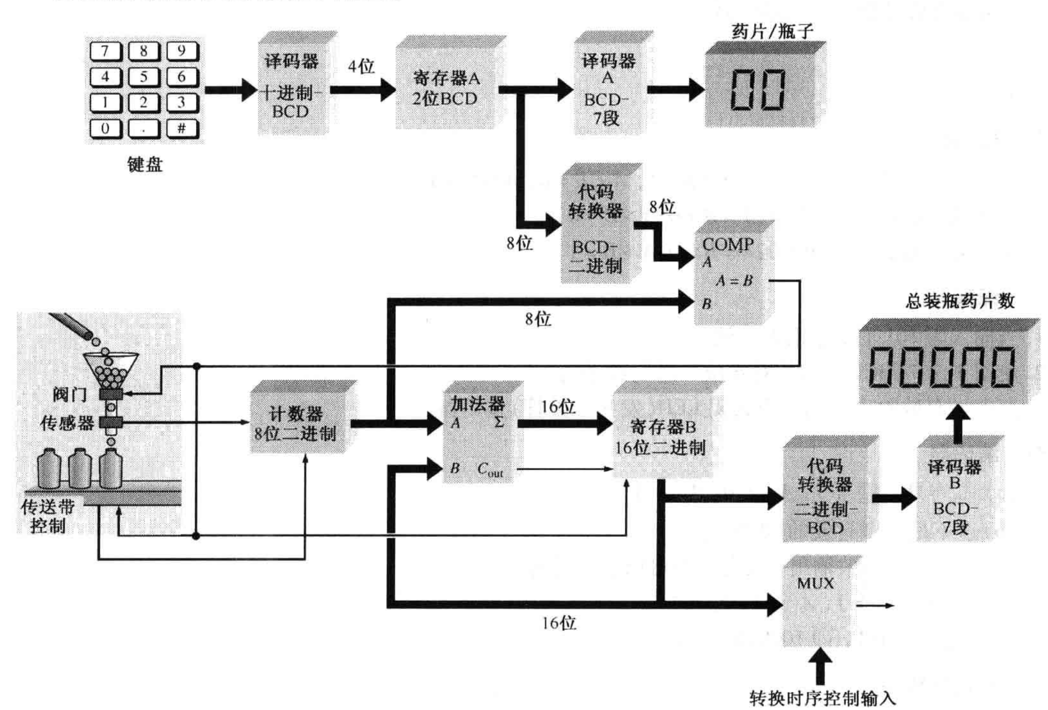


图 6.83

41. 使用74LS85完成图6.83的药片灌装系统的比较器框图，画出包括引脚编号的完整逻辑框图。比较器比较从BCD-二进制转换器得到的8位二进制数(实际仅需要7位数)和从计算机得到的8位二进制数。
42. 在图6.83的药片灌装系统中使用两个BCD-7段译码器。一个用于驱动2位数的药片/瓶子显示器，另一个用于驱动5位数的总装瓶药片数显示器。使用74LS47完成每个译码器，并且画出包括引脚编号的完整逻辑框图。
43. 图6.83的系统框图给出的编码器对每个十进制数的键编码，并且把它们转换为BCD码。使用一个74LS147芯片完成这个功能，画出包括引脚编号的完整的逻辑框图。
44. 图6.83的系统需要两个代码转换器。BCD-二进制转换器把寄存器A中的2位BCD码转换为8位二进制数(实际仅需要7位，因为最高有效位总是0)。使用合适的集成电路代码转换器完成BCD到二进制的转换功能，并且画出包括引脚编号的完整逻辑框图。

答案

温故而知新

6.1节 基本加法器

1. (a) $\Sigma = 1$, $C_{out} = 0$　　(b) $\Sigma = 0$, $C_{out} = 0$
   (c) $\Sigma = 1$, $C_{out} = 0$　　(d) $\Sigma = 0$, $C_{out} = 1$
2. $\Sigma = 1$, $C_{out} = 0$

6.2节 并行二进制加法器

1. $C_{out}\Sigma_4\Sigma_3\Sigma_2\Sigma_1 = 11001$
2. 需要3个74LS283完成两个10位数的相加。

6.3节 异步进位与超前进位加法器

1. $C_g = 0$, $C_p = 1$
2. $C_{out} = 1$

6.4节 比较器

1. 当$A = 1011$和$B = 1010$时，$A > B = 1$, $A < B = 0$, $A = B = 0$
2. 右边的比较器：$A < B = 1$; $A = B = 0$; $A > B = 0$
   左边的比较器：$A < B = 0$; $A = B = 0$; $A > B = 1$

6.5节 译码器

1. 当输入为101时，输出5有效。
2. 使用4个74HC154芯片对6位二进制数译码。
3. 高电平有效输出驱动一个共阴LED(发光二极管)显示器。

6.6节 编码器

1. (a) $A_0 = 1$, $A_1 = 1$, $A_2 = 0$, $A_3 = 1$
   (b)这不是一个有效的BCD码。
   (c)对于一个有效输出，仅有一个可以是有效的。
2. (a) $\overline{A}_3 = 0$, $\overline{A}_2 = 1$, $\overline{A}_1 = 1$, $\overline{A}_0 = 1$
   (b)输出是0111，即1000(8)的反码。

6.7节 代码转换器

1. 10000101(BCD)1010101_2

2. 一个 8 位二进制-格雷码转换器由 7 个异或门组成，它们的排列如图 6.43 所示，但是输入为 $B_0 \sim B_7$。

6.8 节　多路复用器(数据选择器)

1. 输出是 0。
2. (a)74LS157：双 2 输入数据选择器。
   (b)74LS151：8 输入数据选择器。
3. 随着数据选择输入序列按二进制状态变化，数据输出在高低电平之间变化。
4. (a)74LS157 把两个 BCD 码分配到 7 段译码器上。
   (b)74LS47 把 BCD 码译码以点亮显示器。
   (c)74LS139 使得 7 段显示交替变化。

6.9 节　多路分配器

1. 一个译码器可以用做数据选择器，输入线用做数据选择端，使能线(*Enable*)用做数据输入。
2. 除了 D_{10} 为低电平，其他输出全部为高电平。

6.10 节　奇偶发生器/校验器

1. (a)偶校验：1110100　(b)偶校验：001100011
2. (a)奇校验：11010101　(b)奇校验：11000001
3. (a)码正确，4 个 1　(b)码不正确：7 个 1

例题的相关问题

6.1　$\Sigma = 1$，$C_{out} = 1$

6.2　$\Sigma_1 = 0$，$\Sigma_2 = 0$，$\Sigma_3 = 1$，$\Sigma_4 = 1$

6.3　1011 + 1010 = 10101

6.4　参见图 6.84。

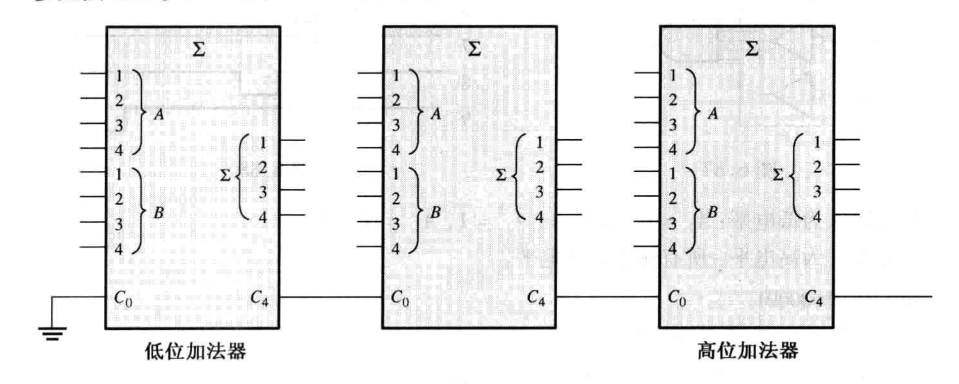


图 6.84

6.5　参见图 6.85。

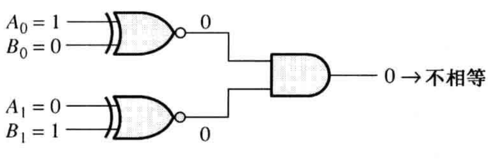


图 6.85

6.6　$A > B = 0$，$A = B = 0$，$A < B = 1$

6.7　参见图 6.86。

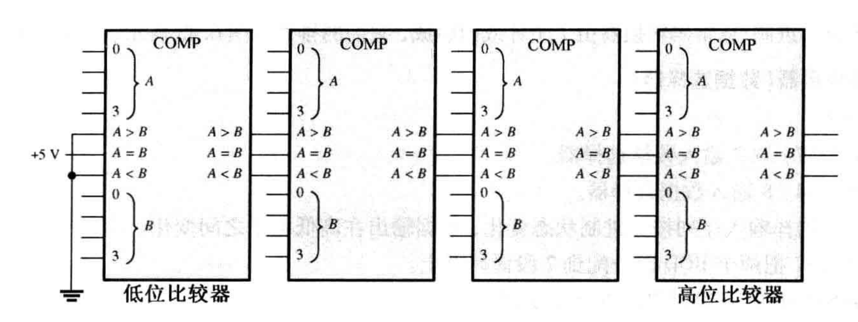


图6.86

6.8 参见图6.87。

6.9 输出22。

6.10 参见图6.88。

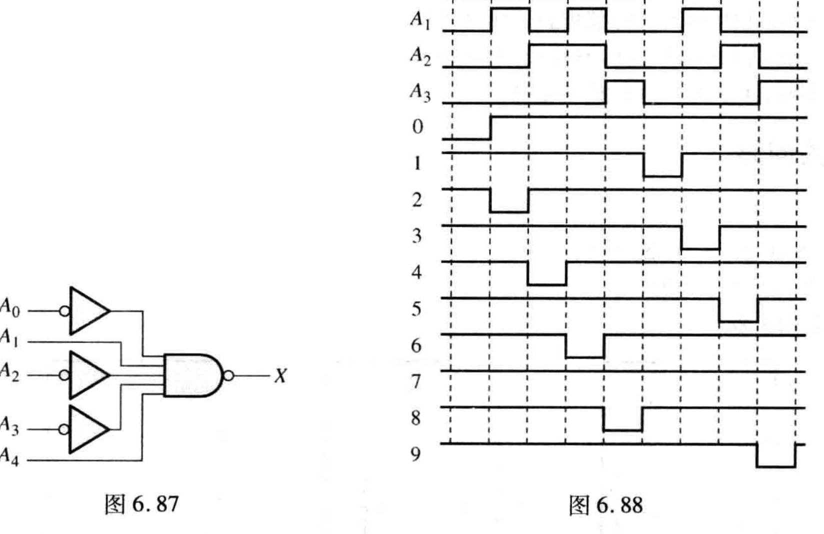


图6.87　　图6.88

6.11 所有输入为低电平：$\overline{A}_0 = 0$，$\overline{A}_1 = 1$，$\overline{A}_2 = 1$，$\overline{A}_3 = 0$。
所有输入为高电平：所有输出为高电平。

6.12 BCD码01000001

→	00000001	1
→	00101000	40
二进制数	00101001	41

6.13 7个异或门。

6.14 参见图6.89。

6.15 D_0: $S_3 = 0, S_2 = 0, S_1 = 0, S_0 = 0$
D_4: $S_3 = 0, S_2 = 1, S_1 = 0, S_0 = 0$
D_8: $S_3 = 1, S_2 = 0, S_1 = 0, S_0 = 0$
D_{13}: $S_3 = 1, S_2 = 1, S_1 = 0, S_0 = 1$

6.16 参见图6.90。

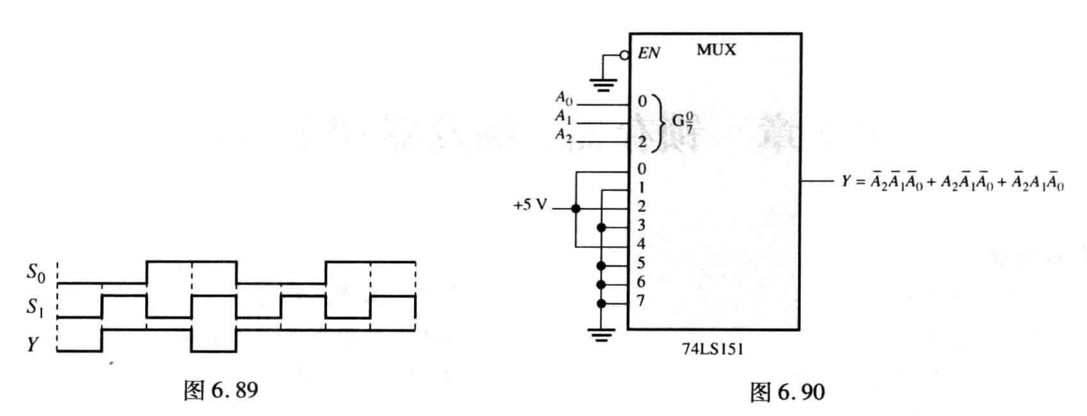


图 6.89　　图 6.90

6.17　参见图 6.91。

6.18　参见图 6.92。

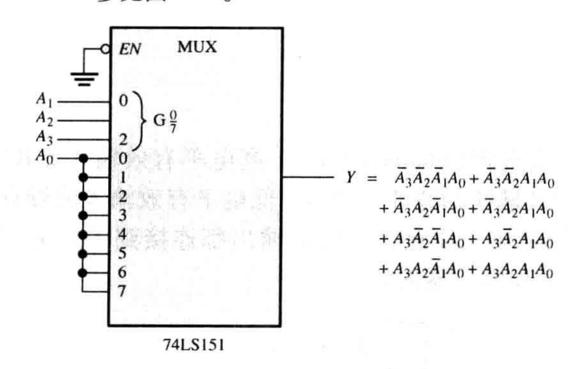


图 6.91

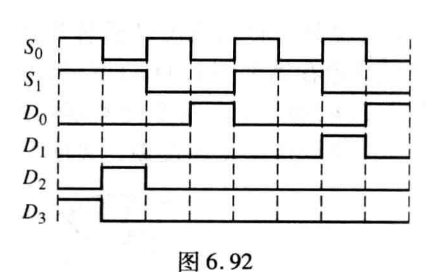


图 6.92

判断题

1. T　2. F　3. T　4. F　5. F　6. T　7. T　8. F　9. T　10. F

自测题

1. (a)　2. (b)　3. (c)　4. (a)　5. (d)　6. (b)
7. (c)　8. (b)　9. (a)　10. (d)　11. (c)　12. (f)

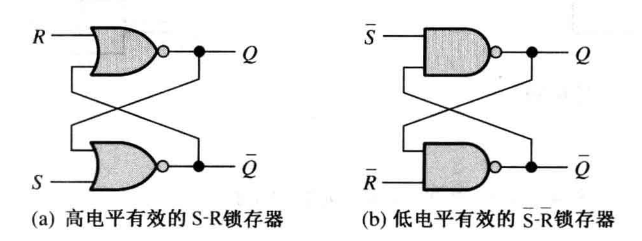
第7章　锁存器、触发器和定时器

章节提纲

7.1　锁存器

7.1.1　S-R(置位-复位)锁存器

锁存器是一种双稳态逻辑芯片或者多谐振荡器(multivibrator)。高电平有效输入S-R(置位-复位)锁存器如图7.1(a)所示，由两个交叉耦联的或非门组成；低电平有效输入的锁存器如图7.1(b)所示，由两个交叉耦联的与非门组成。注意每个门的输出都连接到了对面门的输入上。这就产生了正反馈，这是所有锁存器和触发器的特征。



(a) 高电平有效的S-R锁存器　(b) 低电平有效的$\bar{S}$-$\bar{R}$锁存器

图7.1　S-R锁存器的两个版本。打开文件F07-01(a)和(b)，观察两者的运行结果



计算机小知识

锁存器有时在计算机系统中用来向总线上多路传输数据。例如，来自外部数据源输入计算机的数据不得不和其他地方的数据源共享数据总线。当数据总线对于外部数据源变得不可用时，现存的数据就必须临时存储起来，而安置在外部数据源和数据总线之间的锁存器就可以执行这个任务。当数据总线对于外部数据源不可用时，锁存器就必须从总线上断开，使用一种称为三态的方法。当数据总线可用时，外部数据就会通过锁存器，因此称为透明锁存器。门控D锁存器完成这个功能，因为当它开启时，其输入上的数据就会出现在输出上，就像直接连接一样。一旦锁存器关闭，输入上的数据就存储起来。

为了解释锁存器的运算，使用图7.1(b)中的与非门$\bar{S}$-$\bar{R}$锁存器。使用用于表示与非门的非-或等价符号，把这个锁存器重新绘制在图7.2中。这样做是因为在$\bar{S}$和$\bar{R}$线上是低电平有效输入。

图7.2的锁存器具有两个输入$\bar{S}$和$\bar{R}$，以及两个输出Q和$\bar{Q}$。假设这两个输入和Q输出都

是高电平，这是锁存器通常的状态。由于 Q 输出连接到门 G_2 的一个输入上，而 $\bar{R}$ 输入为高电平，所以 G_2 的输出必定为低电平。这个低电平输出回接耦联到门 G_1 的一个输入上，以保证它的输出为高电平。

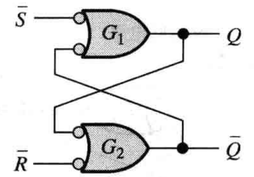


图 7.2　图 7.1(b)中的与非门锁存器的非-或等价图

◇ 锁存器可以保持在两种状态的任何一种状态下，即置位或复位。

当 Q 输出为高电压时，锁存器就处于置位(SET)状态。这个状态将一直保持下去，直到低电平短暂地加在 $\bar{R}$ 输入上。在 $\bar{R}$ 输入为低电平、$\bar{S}$ 输入为高电平时，门 G_2 的输出就被迫为高电平。位于 $\bar{Q}$ 输出上的这个高电平回接耦联到 G_1 的一个输入上，而 $\bar{S}$ 输入为高电平，所以 G_1 的输出就变为低电平。然后，Q 输出上的这个低电平又回接耦联到 G_2 的一个输入上，从而保证 Q 输出保持为高电平，即使 $\bar{R}$ 输入上的低电平移走，Q 输出仍然为高电平。当 Q 输出为低电平时，锁存器就处于复位(RESET)状态。这时锁存器将一直保持在复位状态直至低电平加在 $\bar{S}$ 输入上。

在一般的运算中，锁存器的输出总是互为反相的。

◇ 置位的意思是 Q 输出为高电平。

◇ 复位的意思是 Q 输出为低电平。

当 Q 为高电平时，$\bar{Q}$ 为低电平；而当 Q 为低电平时，$\bar{Q}$ 为高电平。

当低电平同时加在 $\bar{S}$ 和 $\bar{R}$ 时，低电平有效输入 $\bar{S}$-$\bar{R}$ 锁存器的运算中就会发生无效情况。只要低电平是同时加在输入上，Q 和 $\bar{Q}$ 输出就被迫为高电平，因此违反了输出基本上互为反相的运算。同样，如果低电平同时被释放，两个输出将趋向于变为低电平。由于门的传输延迟时间总会有一些小差别，其中有一个门在转换中占据优势，它的 1 输出变为 0。反过来，这就会迫使较慢门的输出保持为高电平。在这种情况下，就不能准确地预知锁存器的下一个状态。

图 7.3 解释了在输入电平的 4 种可能组合的每一种情况下，低电平有效输入 $\bar{S}$-$\bar{R}$ 锁存器的运算(前三个组合是有效的，但是最后一个是无效的)。表 7.1 以真值表的形式总结了逻辑运算。图 7.1(a) 中高电平有效输入的或非门的运算也一样，但是需要使用相反的逻辑电平。

高电平有效输入和低电平有效输入锁存器的逻辑符号如图 7.4 所示。

例 7.1 解释了低电平有效输入 $\bar{S}$-$\bar{R}$ 锁存器如何对输入的情况做出响应。低电平脉冲以某种顺序加到每个输入上，结果观察到 Q 的输出波形。避免 $\bar{S}=0$、$\bar{R}=0$ 的情况，因为这导致了一个无效的运算模式，并且这是任何置位-复位锁存器的一个主要缺陷。

例 7.1　如果图 7.5(a)的 $\bar{S}$ 和 $\bar{R}$ 波形应用于图 7.4(b)中锁存器的输入上，确定在 Q 输出上将会观察到的波形。假设 Q 初始值为低电平。

解：参见图 7.5(b)。

相关问题[①]：如果图 7.5(a)被反相后再加在输入上，确定高电平有效输入 S-R 锁存器的 Q 输出。

① 答案在本章的结尾。

瞬间低电平

当 $\overline{S}$ 变为低电平时，输出发生变化；当 $\overline{S}$ 变回高电平后，状态保持

因为锁存器已经置位，没有发生变化

锁存器开始复位 ($Q=0$)　　锁存器开始置位 ($Q=1$)

(a) 两种可能的置位

当 $\overline{R}$ 变为低电平时，输出发生变化；当 $\overline{R}$ 变回高电平后，状态保持

因为锁存器已经复位，没有发生变化

锁存器开始置位 ($Q=1$)　　锁存器开始复位 ($Q=0$)

(b) 两种可能的复位

如果事先已经置位，输出状态没有发生变化，锁存器保持置位状态；如果事先已经复位，保持复位状态

输入回到高电平后，状态不确定

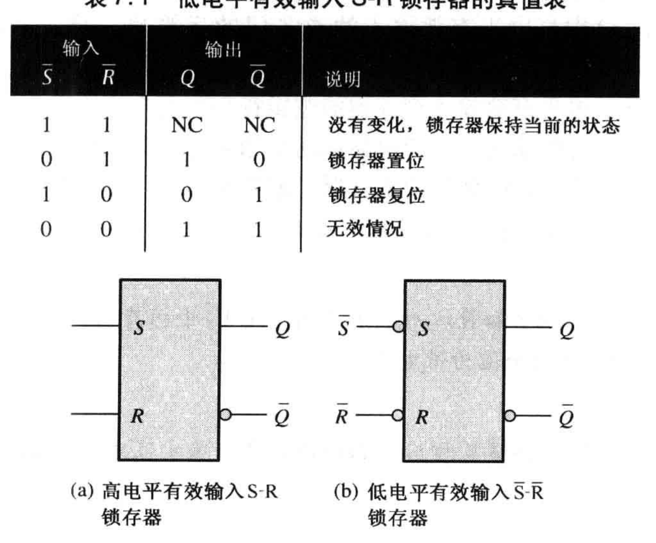
输入都为高电平　　两个输入同时为低电平

(c) 没有变化的情况　　(d) 无效的情况

图 7.3　$\overline{S}$-$\overline{R}$ 锁存器运算的三种模式(置位，复位，没有变化)和无效情况

表 7.1　低电平有效输入 $\overline{S}$-$\overline{R}$ 锁存器的真值表

输入		输出		
$\overline{S}$	$\overline{R}$	Q	$\overline{Q}$	说明
1	1	NC	NC	没有变化，锁存器保持当前的状态
0	1	1	0	锁存器置位
1	0	0	1	锁存器复位
0	0	1	1	无效情况



(a) 高电平有效输入S-R锁存器　　(b) 低电平有效输入 $\overline{S}$-$\overline{R}$ 锁存器

图 7.4　S-R 和 $\overline{R}$-$\overline{S}$ 锁存器的逻辑符号

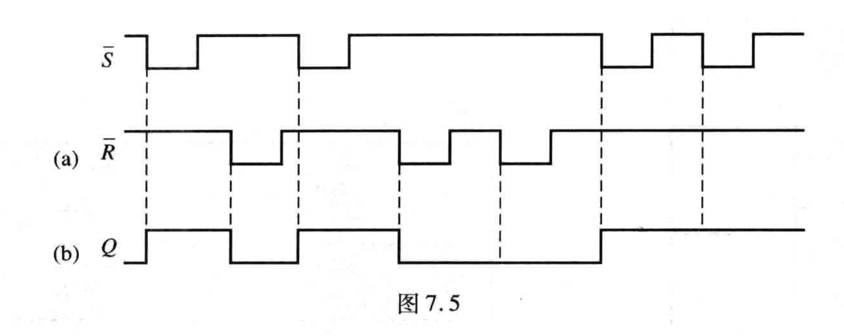


图 7.5

7.1.2　应用举例

锁存器作为触点抖动消除器　$\overline{S}$-$\overline{R}$ 锁存器的一个很好的应用例子是消除机械开关接触“抖动”。当开关的触点和开关闭合处的接触面撞击时，会发生几次物理振动或抖动，然后才能形成最后的固定接触。虽然这些抖动的持续时间很短，但是它们产生电压尖脉冲，这些电压尖脉冲在数字系统中常常是不可接受的。这种情况如图 7.6(a)所示。

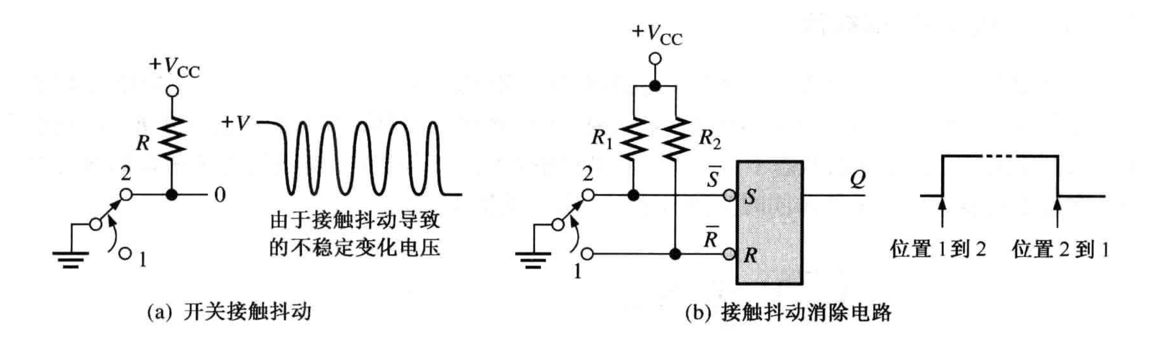


图 7.6　$\overline{S}$-$\overline{R}$ 锁存器用以消除开关接触抖动

$\overline{S}$-$\overline{R}$ 锁存器可以用来消除开关抖动的影响，如图 7.6(b)所示。这个开关一般处在位置 1，保持 $\overline{R}$ 输入为低电平和锁存器复位。当开关合向位置 2 时，由于上拉电阻连接 V_{CC}，$\overline{R}$ 就变为高电平，开关闭合的第一次接触 $\overline{S}$ 变为低电平。尽管在开关抖动之前，$\overline{S}$ 在低电平上仅仅保持了很短的时间，但是这点时间足以使锁存器置位。此后，由于开关抖动在 $\overline{S}$ 输入上产生的任何电压尖脉冲不会影响锁存器，并且保持为置位状态。注意锁存器的 Q 输出提供了从低电平到高电平的净变化，因此消除了由于触点抖动产生的电压尖脉冲。类似地，当开关拨回到位置 1 时，就会产生从高电平到低电平的净变化。



74LS279A 置位-复位锁存器

74LS279A 是一个四 $\overline{S}$-$\overline{R}$ 锁存器，如图 7.7(a)的逻辑框图所示，引脚图如图 7.7(b)所示。注意 4 个锁存器中的两个分别有两个 $\overline{S}$ 输入。

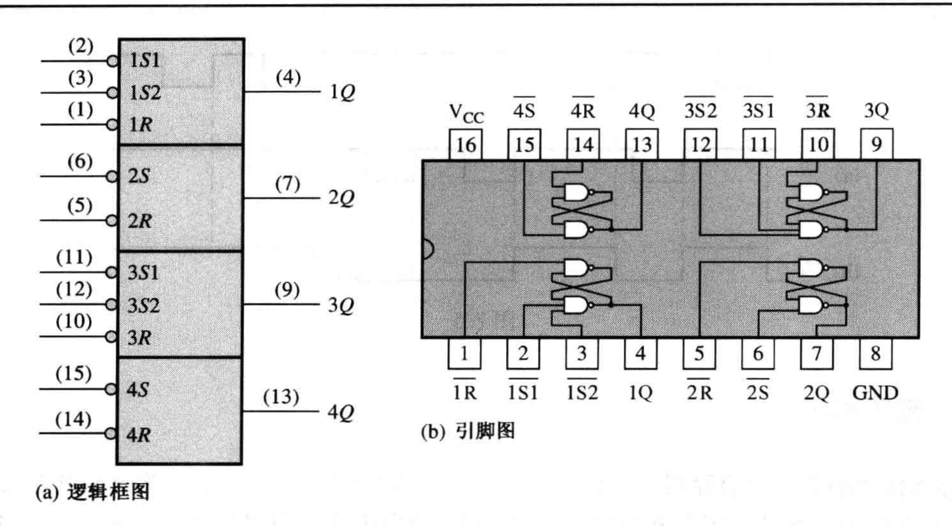


图 7.7 74LS279A 四 $\overline{S}$-$\overline{R}$ 锁存器

7.1.3 门控 S-R 锁存器

门控锁存器需要一个使能输入 *EN*(*G* 也用来表示使能输入)。门控锁存器的逻辑框图和逻辑符号如图 7.8 所示。当高电平加在 *EN* 输入时, *S* 和 *R* 输入控制锁存器的状态。在 *EN* 的高电平没有到来之前, 锁存器的状态不会改变; 但是只要 *EN* 保持高电平, 输出就由 *S* 和 *R* 的状态控制。在这个电路中, 当 *S* 和 *R* 同时为高电平时, 就会发生无效状态。

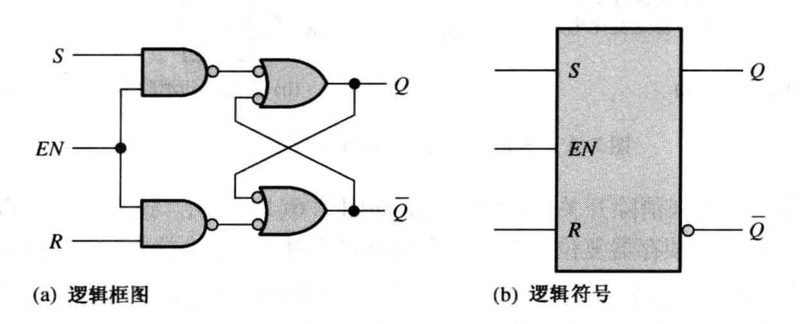


图 7.8 门控 S-R 锁存器

例 7.2 如果图 7.9(a)所示的输入加到初始状态为复位的门控 S-R 锁存器, 确定 *Q* 输出波形。

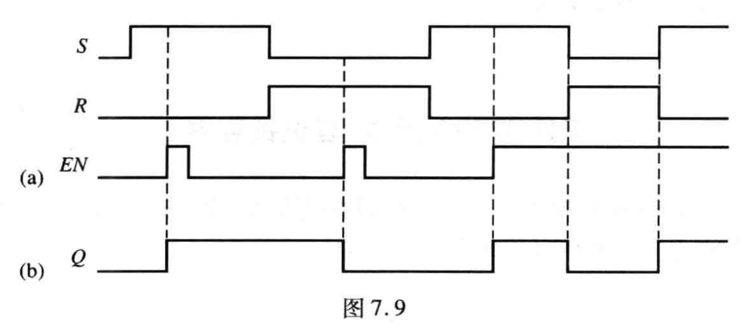


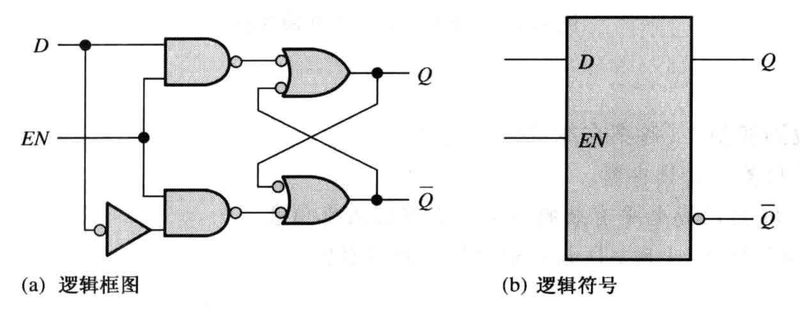
图 7.9

解: Q 输出波形如图 7.9(b)所示。当 S 为高电平和 R 为低电平时，EN 上的高电平输入就使锁存器置位。当 S 为低电平和 R 为高电平时，EN 上的高电平输入就使锁存器复位。当 S 和 R 都为低电平时，Q 输出的状态不会变化。

相关问题: 如图 7.9(a)所示，如果 S 和 R 输入反相，确定门控 S-R 锁存器的 Q 输出。

7.1.4 门控 D 锁存器

另一种类型的门控锁存器称为 D 锁存器。它不同于 S-R 锁存器，因为它除了 EN 之外只有一个输入。这个输入称为 D(数据)输入。图 7.10 给出了 D 锁存器的逻辑框图和逻辑符号。当 D 输入为高电平和 EN 输入为高电平时，锁存器就会置位。当 D 输入为低电平而 EN 为高电平时，锁存器就会复位。使用另外一种方式表述为，当 EN 为高电平时，输出 Q 跟随输入 D。



(a) 逻辑框图　　(b) 逻辑符号

图 7.10 门控 D 锁存器

例 7.3 如果图 7.11(a)给出的输入加在初始状态为复位的门控 D 锁存器，确定 Q 输出波形。

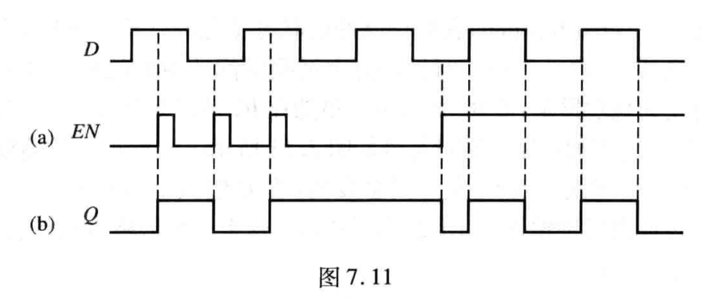


图 7.11

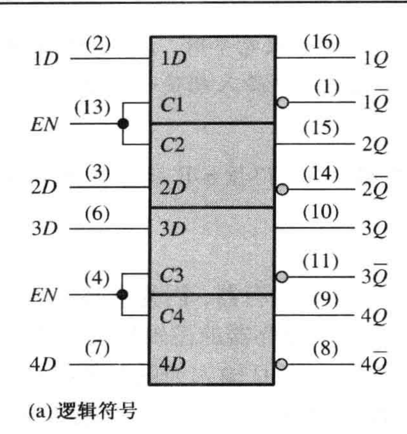
解: Q 输出波形如图 7.11(b)所示。当 D 为高电平和 EN 为高电平时，Q 就会变为高电平。当 D 为低电平和 EN 为高电平时，Q 就会变为低电平。当 EN 为低电平时，锁存器的状态不受 D 输入的影响。

相关问题: 如果图 7.11(a)的 D 输入反相，确定该门控 D 锁存器的 Q 输出。



74LS75 D 锁存器

门控锁存器的一个例子是 74LS75，由图 7.12(a)的逻辑符号表示。这个芯片具有 4 个锁存器。注意每一个高电平有效 EN 输入都由两个锁存器共享，并且命名为控制输入(C)。每个锁存器的真值表如图 7.12(b)所示。真值表中的 X 表示“无关”的情况。在这种情况下，当 EN 输入为低电平时，D 输入就无关紧要了，这是因为输出并没有受到影响，并且保持先前的状态。



(a)逻辑符号

输入		输出		
D	EN	Q	$\overline{Q}$	说明
0	1	0	1	复位
1	1	1	0	置位
X	0	Q_0	$\overline{Q}_0$	没有变化

注：Q_0是在指明的输入条件建立之前的输出电平。

(b)真值表（每个锁存器）

图 7.12 74LS75 4 门控 D 锁存器

7.1 节 温故而知新 （答案在本章的结尾。）

1. 列出 3 种类型的锁存器。
2. 为图 7.1(a)的高电平有效的 S-R 锁存器写出真值表。
3. 当 $EN=1$ 和 $D=1$ 时，D 锁存器的输出是什么？

7.2 边沿触发器

◇ 动态输入指示器▷的意思是触发器的状态变化仅发生在时钟脉冲的边沿。

边沿触发器(edge-triggered flip-flop)在时钟脉冲的状态变化时，发生在时钟脉冲的正边沿(上升沿)或者负边沿(下降沿)，并且只有在时钟的这个转换瞬间才对它的输入做出响应。本节介绍的 3 种类型的边沿触发的触发器是：S-R，D，J-K。虽然在 IC 系列中没有 S-R 触发器，但它是 D 和 J-K 触发器的基础。所有这些触发器的逻辑符号如图 7.13 所示。注意每一种类型都可以是正边沿触发的(在 C 输入上没有小圆圈)或者是负边沿触发的(在 C 输入上有小圆圈)。通过逻辑符号识别边沿触发器的关键是框内时钟输入(C)上的小三角形。这个三角形称为动态输入指示器。

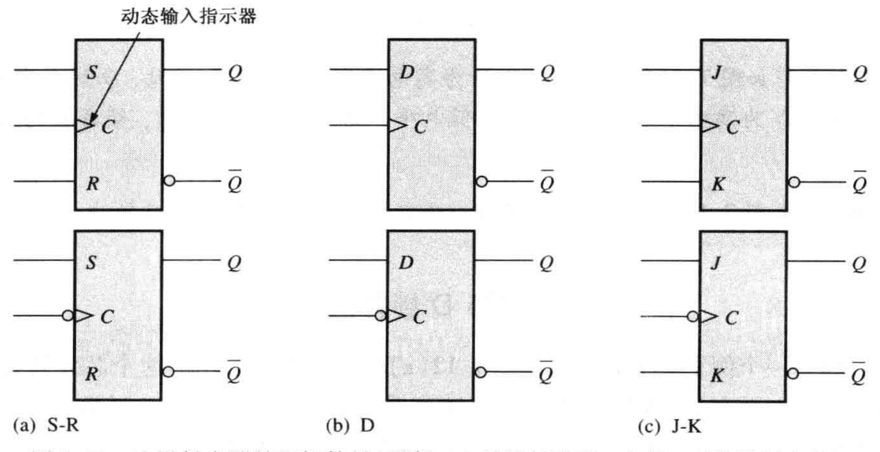


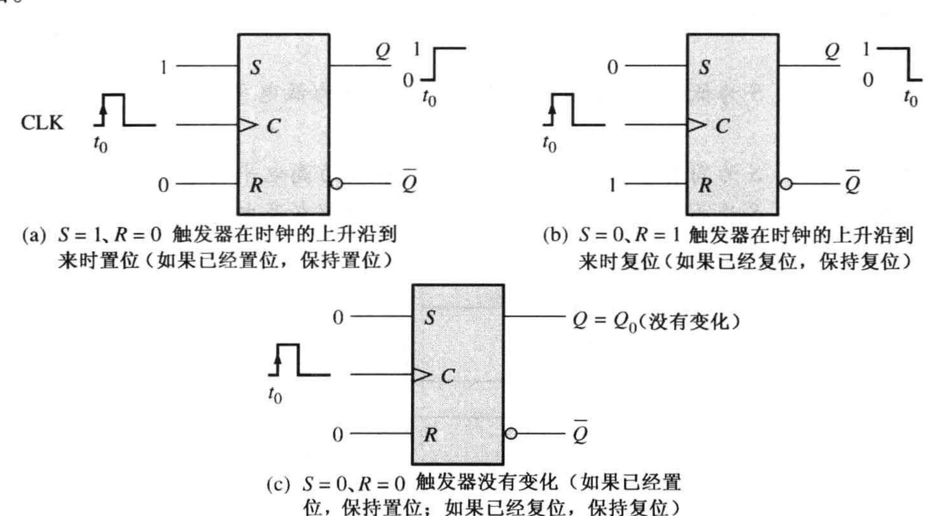
图 7.13 边沿触发器的逻辑符号(顶部：上升沿触发的；底部：下降沿触发的)

7.2.1　边沿触发的 S-R 触发器

◇ S-R 触发器的 S 和 R 输入不能同时为高电平。

S-R 触发器的 S 和 R 输入称为同步输入，因为这两个输入上的数据只能在时钟脉冲的触发边沿传送到触发器的输出。当 S 为高电平和 R 为低电平时，Q 输出在时钟脉冲的触发边沿上变为高电平，且触发器被置位。当 S 为低电平和 R 为高电平时，Q 输出在时钟脉冲的触发边沿上变为低电平，且触发器被复位。当 S 和 R 都是低电平时，输出先前的状态不会改变。在有效的时钟脉冲边沿到来、S 和 R 都是高电平时，就存在无效的情况。

上升沿触发的触发器的基本运算如图 7.14 所示，表 7.2 是这类触发器的真值表。记住，在时钟脉冲的触发边沿之外触发器的状态不会改变。在时钟输入是低电平或高电平时，S 和 R 输入在任何时间(除了在时钟触发转换瞬间周围的一个很短的时间间隔内)都可以改变，但并不影响输出。



(a) $S=1$、$R=0$ 触发器在时钟的上升沿到来时置位(如果已经置位，保持置位)

(b) $S=0$、$R=1$ 触发器在时钟的上升沿到来时复位(如果已经复位，保持复位)

(c) $S=0$、$R=0$ 触发器没有变化(如果已经置位，保持置位；如果已经复位，保持复位)

图 7.14　上升沿触发的 S-R 触发器的运算

表 7.2　上升沿触发的 S-R 触发器的真值表

输入			输出		
S	R	CLK	Q	$\overline{Q}$	说明
0	0	X	Q_0	$\overline{Q}_0$	没有变化
0	1	↑	0	1	置位
1	0	↑	1	0	复位
1	1	↑	?	?	无效情况

↑ = 时钟从低电平到高电平的转换

X = 不相关(“无关”)

Q_0 = 时钟转换之前的输出电平

除了时钟脉冲的下降沿是触发边沿之外，下降沿触发的 S-R 触发器的运算和真值表与上升沿触发的芯片一样。



计算机小知识

计算机中半导体存储器由大量的独立存储单元组成。每个存储单元都保存一个1或者0。存储器的一种类型是静态随机存储器(SRAM)，其中每个单元都使用触发器，因此只要加上直流电源，触发器就能不确定地保持两个状态之一，由此得到静态这个词。这种类型的存储器被归类为易失性存储器，因为当电源断开时，存储的所有数据都会丢失。另一种类型的存储器为动态随机存储器(DRAM)，它使用电容而不是触发器作为基本存储器件，并且必须周期性地刷新以保持存储的数据。

例7.4 对于图7.16(a)中的S、R和CLK输入，确定图7.15中触发器的Q和$\overline{Q}$的输出波形。假设该上升沿触发器的初始值状态为复位。

解： 1. 在时钟脉冲1处，S为低电平，R为低电平，所以Q不会发生变化。

2. 在时钟脉冲2处，S为低电平，R为高电平，所以Q保持为低电平(复位)。
3. 在时钟脉冲3处，S为高电平，R为低电平，所以Q变为高电平(置位)。
4. 在时钟脉冲4处，S为低电平，R为高电平，所以Q变为低电平(复位)。
5. 在时钟脉冲5处，S为高电平，R为低电平，所以Q变为高电平(置位)。
6. 在时钟脉冲6处，S为高电平，R为低电平，所以Q停留在高电平。

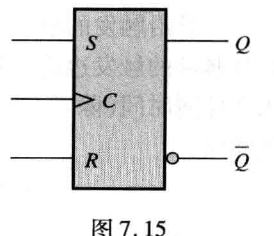


图7.15

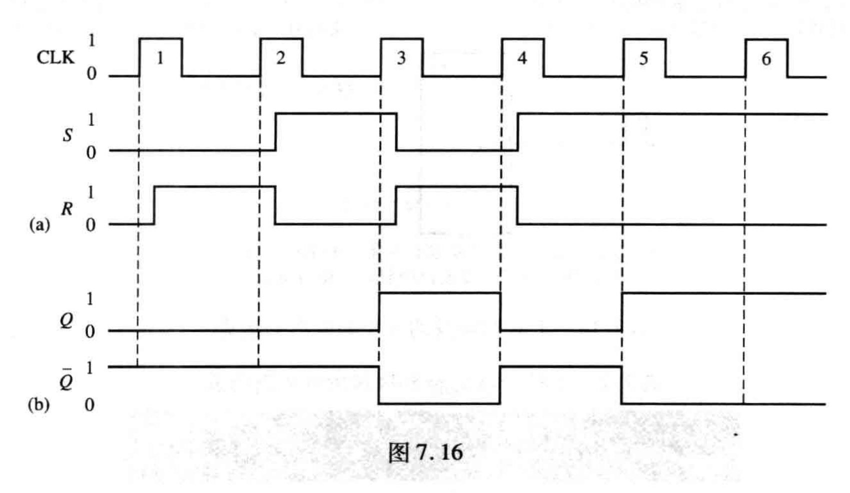


图7.16

一旦Q被确定下来，很容易得到$\overline{Q}$，因为$\overline{Q}$只是Q的反码。Q和$\overline{Q}$的结果波形如图7.16(b)所示，输入波形在图7.16(a)中。

相关问题： 如果该触发器为下降沿触发的芯片，为图7.16(a)所示的S和R输入确定Q和$\overline{Q}$。

7.2.2 边沿触发的一种方法

边沿触发的S-R触发器的简化实现方法如图7.17(a)所示，该图还用来展示边沿触发的概念。介绍S-R触发器并不意味着它是最重要的类型。实际上，D触发器和J-K触发器在IC系列中经常出现，并且比S-R类型应用得更加普遍。然而，理解S-R是很重要的，这是因为D和J-K

触发器是由 S-R 触发器衍生的。注意，S-R 触发器和门控 S-R 锁存器的唯一区别，是它具有一个脉冲转换检测器。

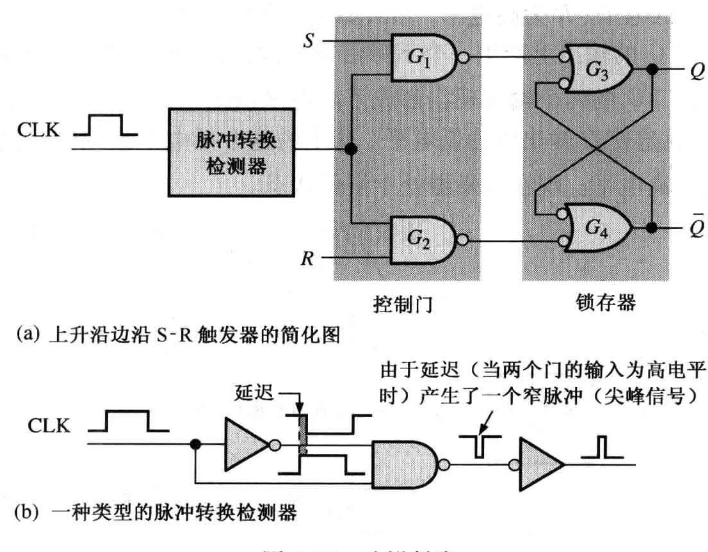


图 7.17 边沿触发

脉冲转换检测器的一种基本类型如图 7.17(b)所示。正如所见，在到达与非门的一个输入时有一个很短的时间延迟，因此反相后的时钟脉冲到达与门输入的时间要比原来的时钟脉冲迟几纳秒。在时钟脉冲的上升沿转换的地方，电路产生了一个持续时间很短的窄脉冲。

在下降沿触发的触发器中，时钟脉冲首先被反相，因此就在下降沿处产生一个窄脉冲。



计算机小知识

所有由硬件执行的逻辑运算也可以在软件中实现。例如，J-K 触发器的运算就可以由具体的计算机指令来完成。如果用两个位表示 J 和 K 输入，那么计算机对于 00 不做任何处理，而对于 10，表示 Q 输出的数据位将会被置位(1)，而对于 01，Q 数据位就会被清零(0)，而对于 11 来说，Q 数据位将会被求反。虽然使用计算机来模拟触发器可能并不常用，但是所强调的是所有的硬件运算都可以用软件来模拟。

图 7.17 中的电路被分成两部分，一部分标记为控制门，另一部分标记为锁存器。控制门引导或者控制时钟窄脉冲进入门 G_3 的输入或者进入门 G_4 的输入，这取决于 S 和 R 输入的状态。为了理解该触发器的运行，开始时假设它处于复位状态($Q=0$)，而 S、R 和 CLK 输入全部为低电平。对于这种情况，门 G_1 和门 G_2 的输出都是高电平。Q 输出上的低电平被耦联回接到门 G_4 的一个输入上，使得其输出 $\bar{Q}$ 为高电平。因为是高电平，G_3 的两个输入都是高电平(记住，门 G_1 的输出为高电平)，这就使得 Q 输出保持为低电平。如果一个脉冲加在 CLK 输入，门 G_1 和 G_2 的输出仍然保持为高电平，这是因为它们被 S 和 R 输入上的低电平给禁止了；所以触发器的状态没有发生改变——它保持在复位状态。

现在使 S 为高电平、R 为低电平，然后加上一个时钟脉冲。由于门 G_1 的 S 输入现在是高电平，并且当 CLK 变为高电平时，门 G_1 的输出会在低电平上持续一个很短的时间(窄脉冲)，这就导致了 Q 输出变为高电平。现在门 G_4 的输入都是高电平(记住，因为 R 为低电平，所以门 G_2 的输出为高电平)，这就迫使 $\bar{Q}$ 输出为低电平。$\bar{Q}$ 上的这个低电平又被耦联回接到门 G_3 的输入上，

将确保 Q 输出保持高电平。触发器现在的状态就是置位。图 7.18 解释了在此情况下发生在触发器内部的逻辑电平的变换。

下一步，使 S 为低电平、R 为高电平，然后加上一个时钟脉冲。因为 R 输入现在为高电平，而时钟的上升沿在门 G_2 的输出上产生一个下降沿窄脉冲，这就使得 $\overline{Q}$ 输出变换为高电平。由于 $\overline{Q}$ 上的这个高电平，门 G_3 的两个输入现在都成了高电平(记住，由于 S 为低电平，所以门 G_1 的输出为高电平)，这就迫使 Q 输出变为低电平。Q 上的这个低电平被耦联回接到门 G_4 的一个输入上，确保 $\overline{Q}$ 保持为高电平。现在触发器处于复位状态。

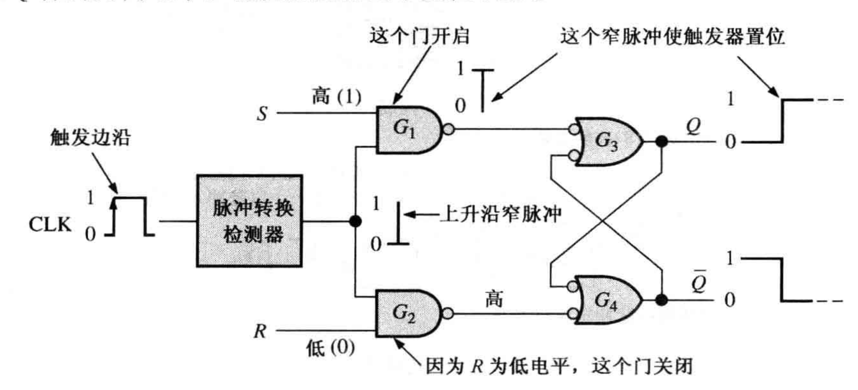


图 7.18　触发器在时钟脉冲的上升沿完成了从复位到置位的转换

图 7.19 解释了在该情况下发生在触发器内部的逻辑电平的转换。和门控锁存器一样，当 S 和 R 同时是高电平，并且有时钟脉冲时，就存在一个无效的情况。这是 S-R 触发器的一个主要缺陷。

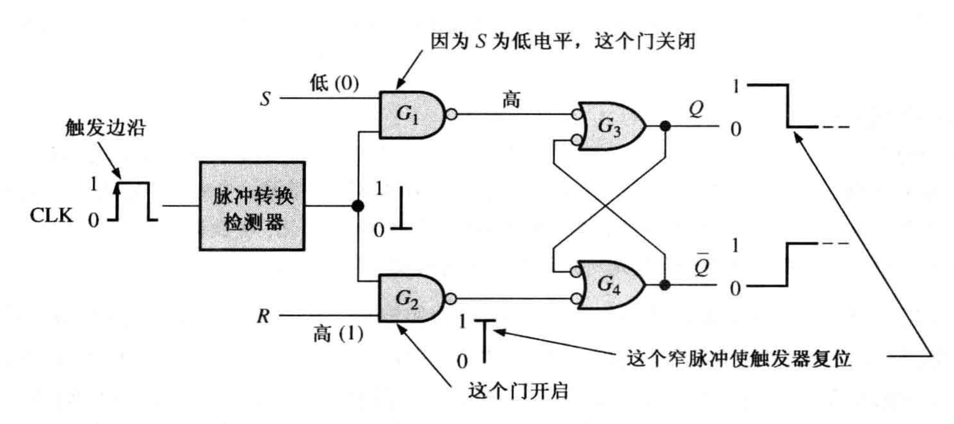


图 7.19　触发器在时钟脉冲的上升沿完成了从置位到复位的转换

7.2.3　边沿触发的 D 触发器

◇ D 触发器的 Q 输出假设了时钟触发边沿的 D 输入状态。

当存储单个数据位(1 或者 0)时，可使用 D 触发器。在 S-R 触发器上加上反相器就形成了基本的 D 触发器，如图 7.20 所示，其中给出了上升沿触发类型。

注意图 7.20 中的触发器除了时钟之外，只有一个输入，即 D 输入。当时钟脉冲到来时，如果 D 输入上有一个高电平，那么触发器就被置位，这样通过时钟脉冲的上升沿，D 输入上的高电平被触发器存储。当时钟脉冲到来时，如果 D 输入上有一个低电平，那么触发器就被复位，

这样通过时钟脉冲的上升沿，D 输入上的低电平被触发器存储。在置位状态下，触发器存储一个 1，而在复位状态下存储一个 0。

上升沿触发的 D 触发器的逻辑运算总结于表 7.3 中。当然，下降沿触发芯片的运算是一样的，除了触发发生在时钟脉冲的下降沿之外。记住，在有效或者触发时钟边沿，Q 跟随 D。

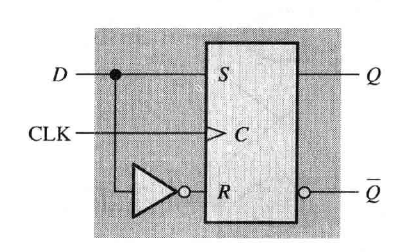


图 7.20　由 S-R 触发器和反相器形成的上升沿D触发器

表 7.3　上升沿触发的 D 触发器的真值表

输入		输出		
D	CLK	Q	$\overline{Q}$	说明
1	↑	1	0	置位(保存1)
0	↑	0	1	复位(保存0)

↑ = 时钟转换低到高

例 7.5　图 7.21(a) 中给出 D 输入和时钟的波形，如果触发器初始状态为复位，确定 Q 输出波形。

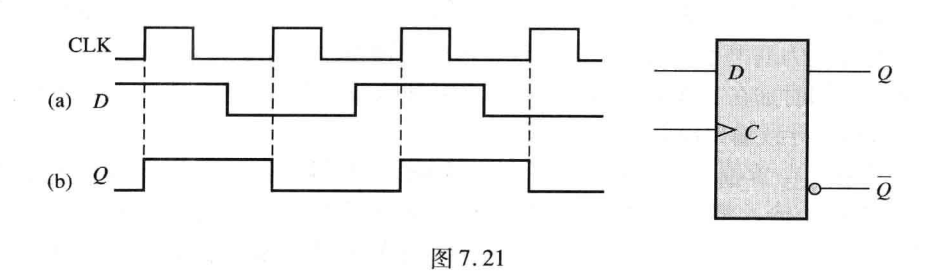


图 7.21

解： 在上升时钟边沿上，Q 输出转换为 D 输入的状态。结果输出如图 7.21(b) 所示。

相关问题： 如果图 7.21(a) 中 D 输入被反相，确定 D 触发器的 Q 输出。

7.2.4　边沿触发的 J-K 触发器

J-K 触发器的用途很多，是广泛应用的触发器类型。在置位、复位和运算的无变化情况方面，J-K 触发器的功能和 S-R 触发器是一样的。区别在于 J-K 触发器没有无效状态而 S-R 触发器具有。

图 7.22 给出了上升沿触发的 J-K 触发器的基本内部逻辑。它和 S-R 边沿触发的触发器的区别在于，Q 输出回接到门 G_2 的输入上。而 $\overline{Q}$ 输出回接到门 G_1 的输入上。这两个控制输入以 Jack Kilby 的名字标记为 J 和 K(Jack Kilby 发明了集成电路)。J-K 触发器也可以有下降沿触发的类型，这种类型中的时钟输入被反相了。

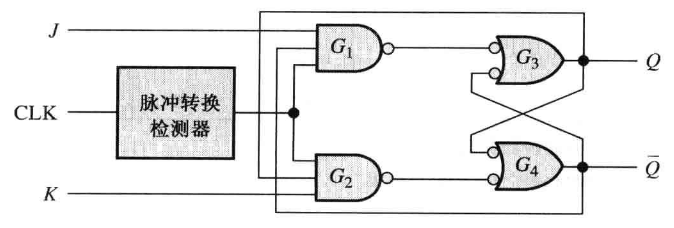


图 7.22　上升沿触发的 J-K 触发器的简化逻辑框图

让我们假设图7.23中的触发器处于复位状态，并且J输入为高电平、K输入为低电平而不是图中所示。当时钟脉冲到来时，由①所指示的上升沿窄脉冲就会通过门G_1，这是因为$\overline{Q}$为高电平并且J也为高电平。这就会导致触发器中的锁存器部分变为置位状态。

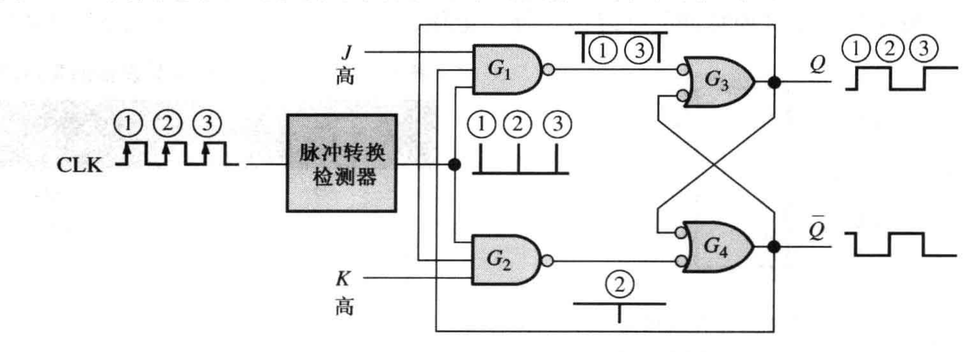


图7.23 当$J=1$和$K=1$时，解释切换操作下的变换

◇ *在切换模式下，J-K触发器在每一个时钟脉冲到来时都改变状态。*

如果使得J为低电平而K为高电平，由②所指示的下一个时钟窄脉冲就会通过门G_2，因为Q为高电平而K也为高电平。这就会导致触发器中的锁存器部分变为复位状态。

如果低电平同时加在J和K输入上，那么当时钟脉冲到来时，触发器就会保持当前的状态。J和K上的低电平导致输出没有变化的情况。

到目前为止，在置位、复位及无变化模式方面，J-K触发器的逻辑运算和S-R触发器是一样的。当J和K输入都是高电平时，运算中的区别就显现了。为了说明问题，假设触发器处于复位状态。$\overline{Q}$上的高电平使得门G_1开启，这样由③所指示的时钟窄脉冲就会通过并使触发器置位。现在Q上有一个高电平，这就允许下一个时钟窄脉冲经过门G_2并使触发器复位。

正如所见，在每一个相继的时钟脉冲上，触发器变为相反的状态。这个模式称为切换(toggle)运算。图7.23解释了触发器处于切换模式时的变换。连接成切换运算的J-K触发器有时候称为T触发器。

表7.4以真值表的形式总结了边沿触发的J-K触发器的逻辑运算。注意这里没有无效状态，而S-R触发器则有无效状态。除了在时钟脉冲的下降沿触发之外，下降沿触发的芯片的真值表和上升沿触发器的真值表是一样的。

表7.4 上升沿触发的J-K触发器的真值表

输入			输出		
J	K	CLK	Q	$\overline{Q}$	说明
0	0	↑	Q_0	$\overline{Q}_0$	没有变化
0	1	↑	0	1	复位
1	0	↑	1	0	置位
1	1	↑	$\overline{Q}_0$	Q_0	切换

↑ = 时钟转换低到高

Q_0 = 时钟转换之前的输出电平

例 7.6　图 7.24(a)所示的波形加在 J、K 和时钟输入。确定 Q 输出，假设该触发器的初始状态为复位。

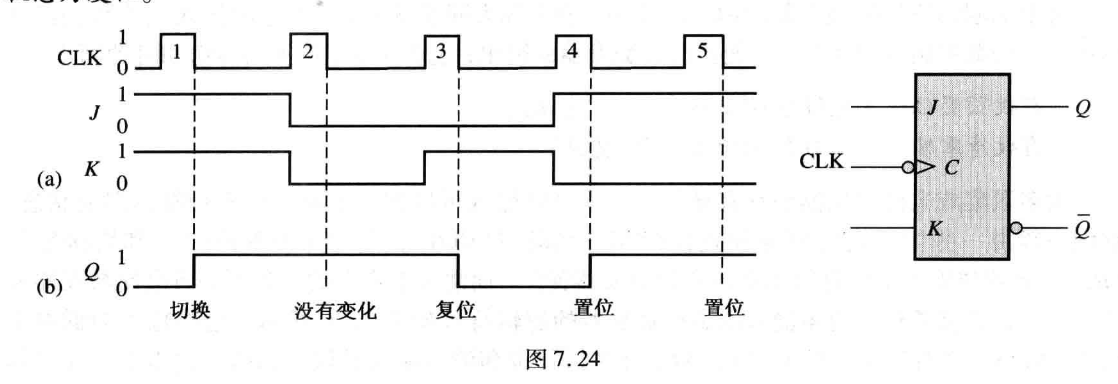


图 7.24

解： 如时钟输入处的小圆圈所示，这时对于一个下降沿触发的触发器，输出 Q 仅在时钟脉冲的下降沿改变。

1. 在第一个时钟脉冲上，J 和 K 都是高电平；并且由于这是切换的情况，Q 就变为高电平。
2. 在时钟脉冲 2 上，输入上出现了无变化的情况，保持 Q 在高电平。
3. 当时钟脉冲 3 到来时，J 为低电平和 K 为高电平，结果就处在复位的情况；Q 变为低电平。
4. 在时钟脉冲 4，J 为高电平和 K 为低电平，结果就处在置位的情况；Q 变为高电平。
5. 当时钟脉冲 5 到来时，J 和 K 仍然处于置位的情况，所以 Q 将保持在高电平。

结果 Q 波形如图 7.24(b)所示。

相关问题： 如果图 7.24(a)中的 J 和 K 输入反相，确定 J-K 触发器的 Q 输出。

例 7.7　图 7.25(a)中的波形加在如图所示的触发器上。确定 Q 输出，假设开始于复位状态。

解： 假设 Q 输出的状态由时钟脉冲的上升沿(触发边沿)的 J 和 K 的输入状态所确定。在时钟触发边沿到来之后，J 和 K 的变化对输出不会产生影响，如图 7.25(b)所示。

相关问题： 交换 J 和 K 的输入，并确定 Q 输出。

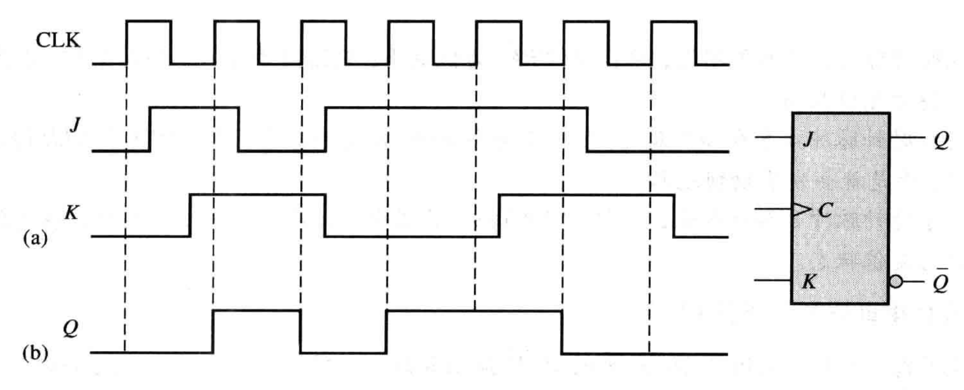


图 7.25

7.2.5 异步预置位输入和清零输入

对于刚刚讨论过的触发器，*S-R*、*D* 和 *J-K* 输入称为同步输入，因为这些输入上的数据仅在时钟脉冲的触发边沿到来时，才能传送到触发器的输出；也就是说，数据的传送和时钟同步。

◇ 有效预置位输入使得 *Q* 输出为高电平(置位)。

◇ 有效清零输入使得 *Q* 输出为低电平(复位)。

大多数集成电路触发器还具有异步输入。这些输入可以独立于时钟而影响触发器的状态。它们一般由一些生产商标记为预置位(PRE)和清零(CLR)，或者直接置位(S_D)和直接复位(R_D)。预置位输入上的有效电平将会使触发器置位，而清零输入上的有效电平将会使触发器复位。一个具有预置位和清零输入的 J-K 触发器的逻辑符号如图 7.26 所示。这些输入为低电平有效，由小圆圈所指示。对于同步运算，这些预置位和清零输入必须都保持在高电平。在正常情况下，预置位和清零不能同时为低电平。

图 7.27 给出了具有低电平有效的预置位($\overline{PRE}$)和清零($\overline{CLR}$)输入的边沿触发的 J-K 触发器的逻辑框图。这个图从根本上解释了这些输入是怎样工作的。正如所看到的那样，图中的连接使得它们无视同步输入 *J*、*K* 和时钟的影响。

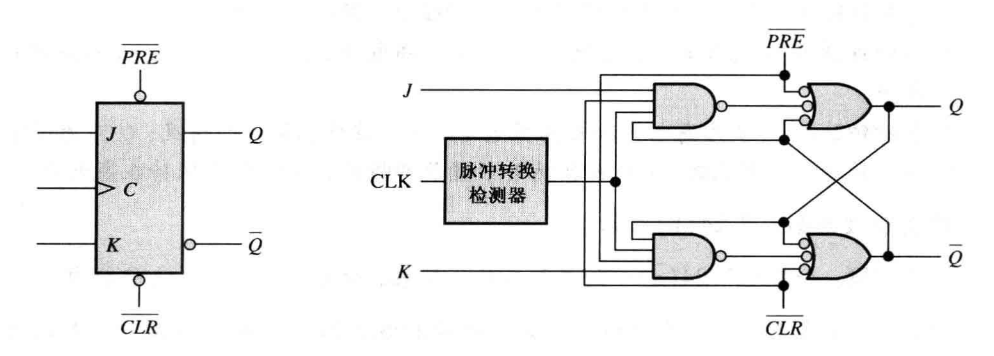


图 7.26 具有低电平有效的预置位和清零输入的J-K触发器的逻辑符号

图 7.27 具有低电平有效的预置位和清零输入的基本J-K触发器的逻辑框图

例 7.8 对于图 7.28 中具有预置位和清零输入的上升沿 J-K 触发器，以及图 7.28(b)的时序图所示的输入，如果 *Q* 初始值为低电平，确定 *Q* 输出。

解：

1. 在时钟脉冲 1、2 和 3 期间，预置位($\overline{PRE}$)为低电平，无论同步输入 *J* 和 *K* 如何，触发器都保持为置位状态。
2. 对于时钟脉冲 4、5、6 和 7 来说，因为 *J* 为高电平，*K* 为高电平，并且 $\overline{PRE}$ 和 $\overline{CLR}$ 都是高电平，于是就会发生切换运算。
3. 对于时钟脉冲 8 和 9 来说，清零($\overline{CLR}$)输入为低电平，无论同步输入怎样，触发器都保持为复位状态。

结果 *Q* 输出如图 7.28(b)所示。

相关问题： 如果交换图 7.28(a)中的 $\overline{PRE}$ 和 $\overline{CLR}$ 波形，那么 *Q* 输出应当是怎样的？

下面来看两个特殊的边沿触发的触发器。它们是 IC 系列中存在的不同触发器类型的代表，并且像大多数其他芯片一样，在 CMOS 和 TTL 逻辑系列中也可以看到。

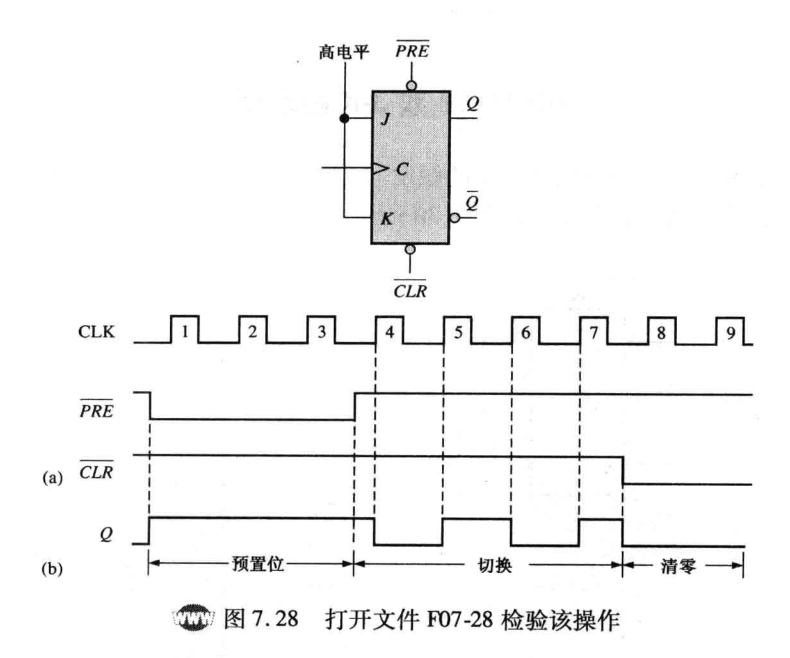


图 7.28　打开文件 F07-28 检验该操作



74AHC74 双 D 触发器

这个 CMOS 芯片包含两个完全一样的 D 触发器，除了共享 V_{CC} 和接地之外都是相互独立的。这两个触发器都是上升沿触发，并且具有低电平有效异步预置位和清零输入。芯片中每个触发器的逻辑符号如图 7.29(a)所示，而表示整个芯片的 ANSI/IEEE 标准单框图符号如图 7.29(b)所示。引脚号位于括号中。

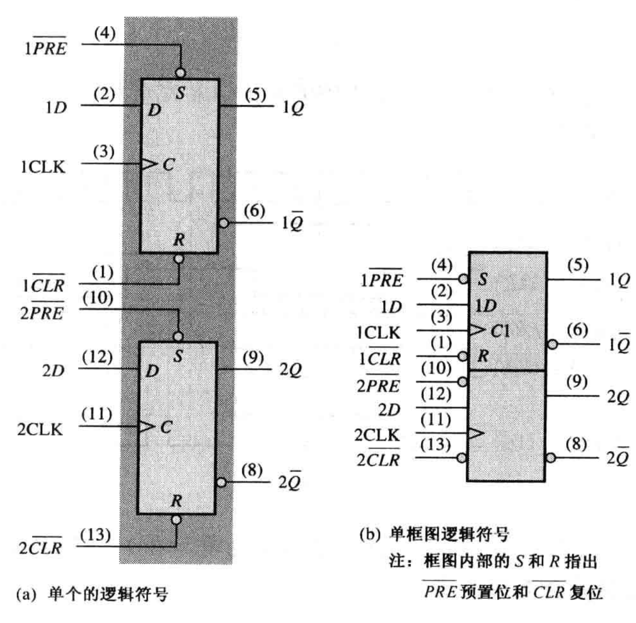
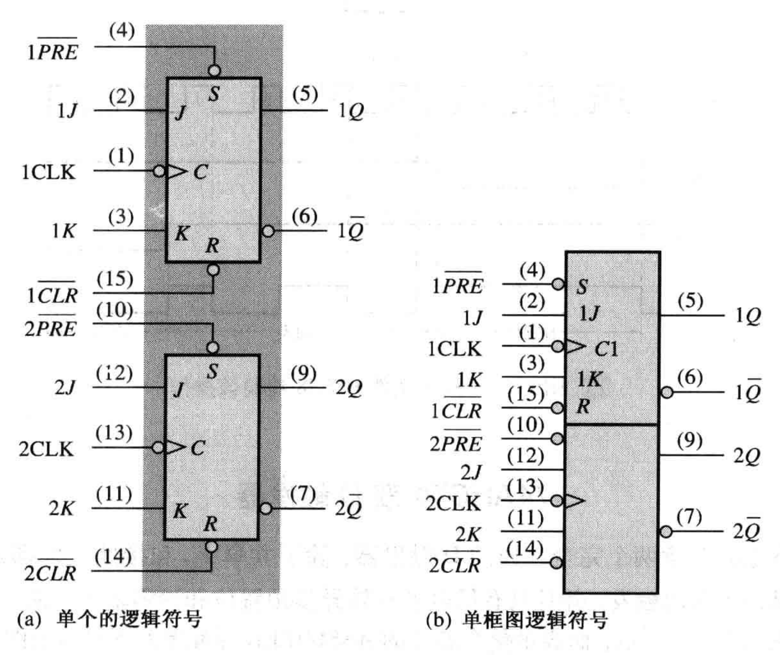


图 7.29　74AHC74 双上升沿 D 触发器的逻辑符号



74HC112 双 J-K 触发器

这种 CMOS 芯片也具有两个完全一样的触发器，它们是下降沿触发的，并且具有低电平有效异步预置位和清零输入。逻辑符号如图 7.30 所示。



(a) 单个的逻辑符号　(b) 单框图逻辑符号

图 7.30　74HC112 双下降沿 J-K 触发器的逻辑符号

例 7.9　图 7.31(a)中的 1*J*、1*K*、1CLK、1 $\overline{PRE}$ 和 1 $\overline{CLR}$ 波形加在 74HC112 芯片中的一个下降沿触发器上。确定 1*Q* 输出波形。

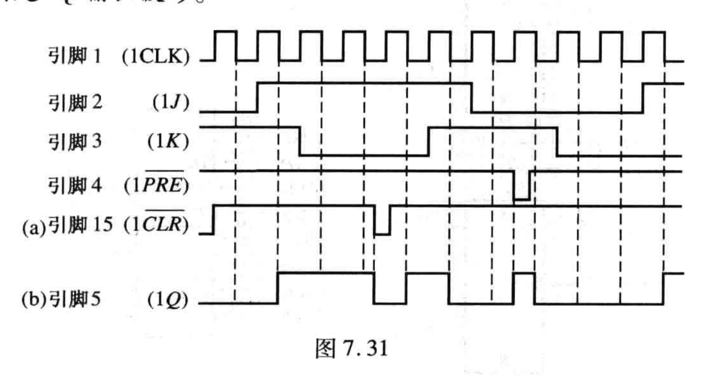


图 7.31

解：结果 1*Q* 波形如图 7.31(b)所示。注意每次低电平加在 1 $\overline{PRE}$ 或者 1 $\overline{CLR}$ 时，无论其他输入的状态如何，触发器就会置位或者复位。

相关问题：如果交换 1 $\overline{PRE}$ 和 1 $\overline{CLR}$ 波形，确定 1*Q* 输出波形。

7.2 节　温故而知新

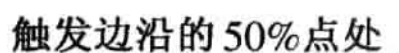
1. 描述门控 S-R 锁存器和 1 边沿触发的 S-R 触发器的区别。
2. J-K 触发器和 S-R 触发器的基本工作原理有什么不同?
3. 假设图 7.21 的触发器是下降沿触发的触发器。对于同样的 CLK 和 D 输入波形，给出输出波形。

7.3　触发器运算特性

7.3.1　传输延迟时间

传输延迟时间是施加输入信号、导致输出发生变化所需要的时间间隔。在触发器运算中，有 4 种传输延迟时间是很重要的:

1. 传输延迟 t_{PLH}，从时钟脉冲的触发边沿到输出的低电平到高电平变换之间所测得的时间。这种延迟如图 7.32(a)所示。
2. 传输延迟 t_{PHL}，从时钟脉冲的触发边沿到输出的高电平到低电平变换之间所测得的时间。这种延迟如图 7.32(b)所示。



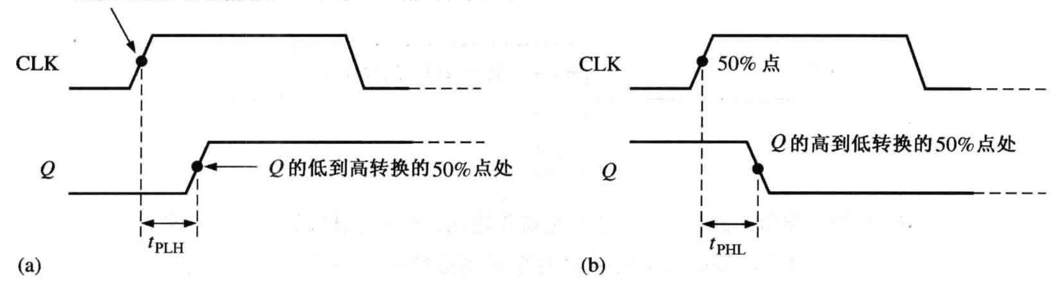


图 7.32　传输延迟，时钟到输出

3. 传输延迟 t_{PLH}，从预置位输入的前沿到输出低电平到高电平的变换之间所测得的时间。这种延迟如图 7.33(a)所示，给出的是低电平有效预置位输入。
4. 传播延迟 t_{PHL}，从清零输入的前沿到输出高电平到低电平的变换之间所测得的时间。这种延迟如图 7.33(b)所示，给出的是高电平有效清零输入。

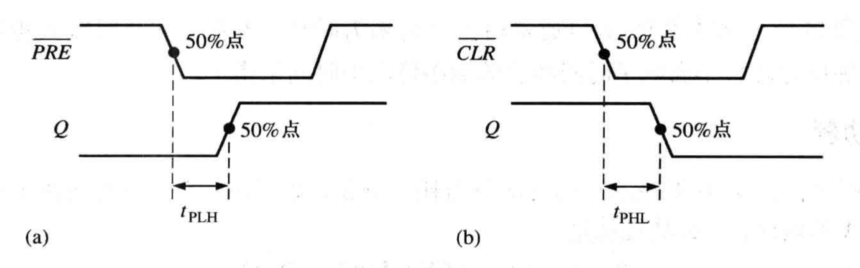


图 7.33　传输延迟，预置位输入到输出和清零输入到输出

7.3.2　建立时间

建立时间(t_s)是输入先于时钟脉冲触发边沿到达所需要的最小时间间隔，在此时间里输入

(J 和 K, S 和 R, 或者 D)的逻辑电平保持不变，这样就使得输入电平可靠地按时序进入触发器。这个时间间隔如图 7.34 中的 D 触发器所示。

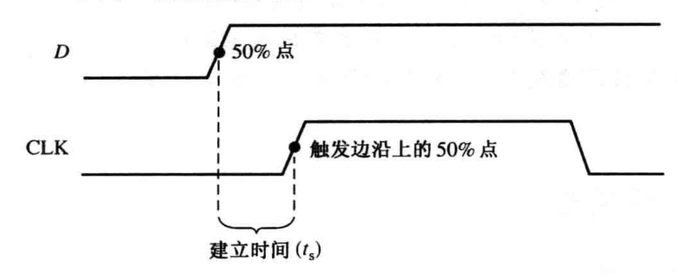


图 7.34 建立时间(t_s)。为了数据可靠进入，在时钟脉冲触发边沿到来之前，D输入上的逻辑电平必须出现的提前时间等于或者大于t_s

7.3.3 保持时间

保持时间(t_h)是在时钟脉冲触发边沿到达之后，输入上的逻辑电平需要保持的最小时间间隔，以使得输入电平可靠地按时序进入触发器。这个时间间隔如图 7.35 中的 D 触发器所示。

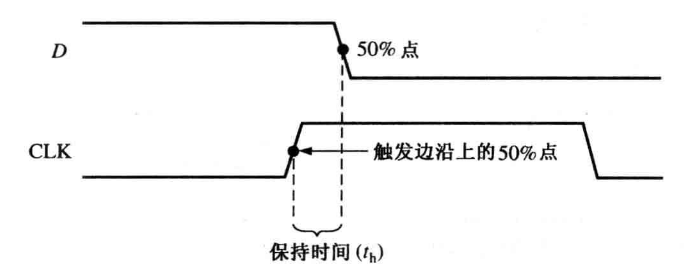


图 7.35 保持时间(t_h)。为了数据可靠进入，在时钟脉冲触发边沿到来之后，D输入上的逻辑电平必须保持的时间等于或者大于t_h

7.3.4 最大时钟频率

最大时钟频率(f_{max})是触发器可以可靠触发的最高速度。对于在最大值之上的时钟频率，触发器将不能足够快地做出响应，并且运算功能也会减弱。

7.3.5 脉冲宽度

可靠运算的最小脉冲宽度(t_W)通常由生产商来为时钟、预置位和清零输入指定。典型的情况是，时钟由它的最小高电平时间和它的最小低电平时间来指定。

7.3.6 功耗

任何数字电路的功耗即该芯片的总功率消耗。例如，如果在 +5 V 直流电源上运行的触发器需要 5 mA 的电流，那么功耗就是

$$P = V_{CC} \times I_{CC} = 5\text{ V} \times 5\text{ mA} = 25\text{ mW}$$

当考虑到直流电源的容量时，功耗在大多数应用中都非常重要。作为一个例子，假设有一个数字系统，总共需要 10 个触发器，并且每一个触发器都耗用 25 mW 的功率。总功率需求就是

$$P_T = 10 \times 25\text{ mW} = 250\text{ mW} = 0.25\text{ W}$$

这就给出了直流电源所需的输出容量。如果触发器运行在一个 +5 V 的直流电源上，那么电源必须提供的电流为

$$I = \frac{250\ \text{mW}}{5\ \text{V}} = 50\ \text{mA}$$

必须使用能够提供至少 50 mA 电流的 +5 V 直流电源。



相应提示

CMOS 相比于 TTL 的一个优点是，它可以运行在一个宽范围的直流电源电压内(典型地为2 V 到 6 V)，所以可以使用没有精确调整的、不太昂贵的电源。同样，可以使用电池作为 CMOS 电路的第二或主要电源。此外，较低的电压就意味着 IC 具有较小的功率损耗。CMOS 的缺陷是它的性能在电源电压较低时会差一些。例如，CMOS 触发器允许的最大时钟频率在 V_{CC} = 2 V 时要比 V_{CC} = 6 V 时低很多。

7.3.7　触发器的具体比较

表 7.5 提供了 4 个同类型的 CMOS 和 TTL 触发器的比较，按照本节所讨论的运行参数。

表 7.5　同类型触发器的 4 个 IC 系列在 25℃时的运行参数的比较

参数	CMOS		双极型(TTL)	
	74HC74A	74AHC74	74LS74A	74F74
t_{PHL} (CLK 到 Q)	17 ns	4.6 ns	40 ns	6.8 ns
t_{PLH} (CLK 到 Q)	17 ns	4.6 ns	25 ns	8.0 ns
t_{PHL} ($\overline{CLR}$ 到 Q)	18 ns	4.8 ns	40 ns	9.0 ns
t_{PLH} ($\overline{PRE}$ 到 Q)	18 ns	4.8 ns	25 ns	6.1 ns
t_s (建立时间)	14 ns	5.0 ns	20 ns	2.0 ns
t_h (保持时间)	3.0 ns	0.5 ns	5 ns	1.0 ns
t_W (时钟高)	10 ns	5.0 ns	25 ns	4.0 ns
t_W (时钟低)	10 ns	5.0 ns	25 ns	5.0 ns
t_W($\overline{CLR}/\overline{PRE}$)	10 ns	5.0 ns	25 ns	4.0 ns
f_{max}	35 MHz	170 MHz	25 MHz	100 MHz
功率，静态	0.012 mW	1.1 mW		
功率，50% 占空比			44 mW	88 mW

7.3 节　温故而知新

1. 定义如下的名词：
   (a) 建立时间　(b) 保持时间
2. 在表 7.5 中哪个具体的触发器可以运行在最高频率？

7.4　触发器应用

7.4.1　并行数据存储

数字系统中的一个共同需求是在一组触发器中同时保存来自并行线的几个数据位。该运算如图 7.36(a)所示，图中使用了 4 个触发器。4 个并行数据线的每一个都连接到了触发器的 D

输入上。触发器的时钟输入连接在一起，使得每一个触发器都由相同的时钟脉冲来触发。在这个例子中，使用了上升沿触发器，所以在时钟的上升沿到来时，D 输入上的数据被触发器同时存储，如图 7.36(b)所示。同样，异步复位(R)输入连接到共同的 $\overline{CLR}$ 线上，用来对所有的触发器进行初始复位。

这 4 个触发器组是用于存储数据的基本寄存器的一个例子。在数字系统中，数据一般以多位一组的形式存储(通常 8 位或者多位一组)，位组可以表示数字、代码或者其他信息。寄存器将在第 9 章得到详细的介绍。

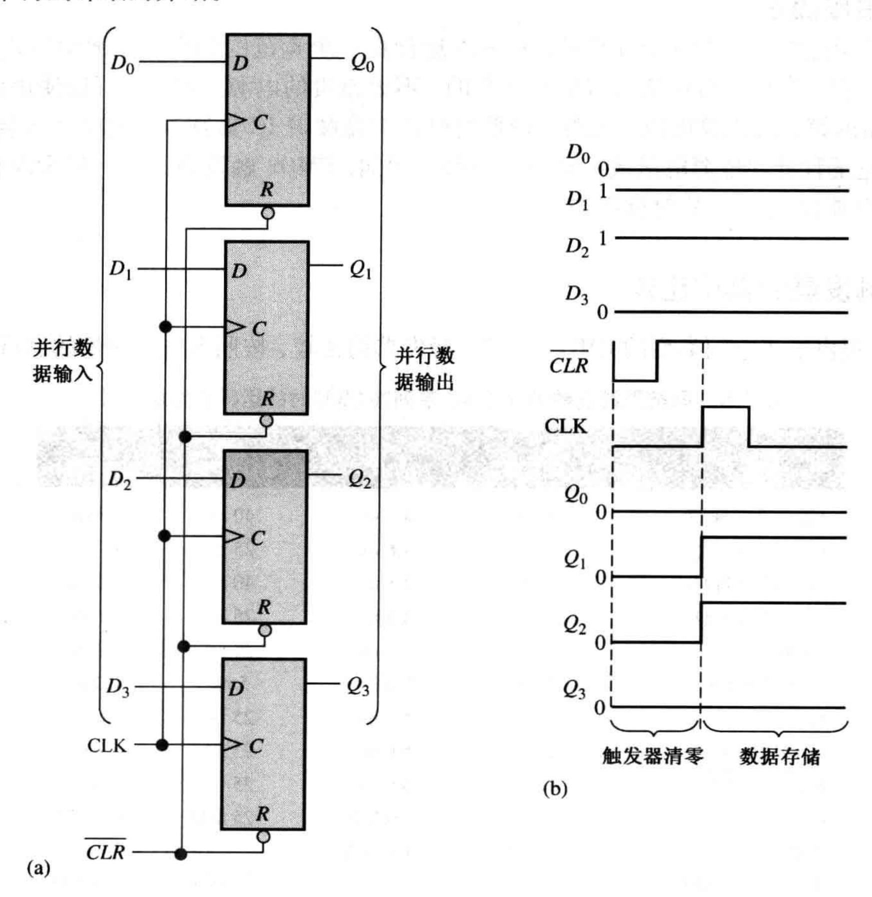


图 7.36 在基本寄存器中用来存储并行数据的触发器例子

7.4.2 分频

触发器的另一个应用是对周期波形的频率进行分(减少)频。当脉冲波形加在一个 J-K 触发器的时钟输入时，J-K 触发器连接成切换状态($J = K = 1$)，这时 Q 输出就是一个频率为时钟输入频率一半的方波。因此，单个触发器可以用做除以 2 芯片，如图 7.37 所示。触发器在每一个触发时钟边沿(在这个例子中是上升沿触发)改变状态。这就产生了一个输出，它的频率变为时钟波形频率的一半。

时钟频率的进一步分频可以通过将触发器的输出用做第二个触发器的时钟输入来实现，如图 7.38 所示。Q_A输出的频率由触发器 Q_B二分频。所以 Q_B输出的频率就是原先时钟输入的 1/4。传输延迟时间没有在时序图中指出。

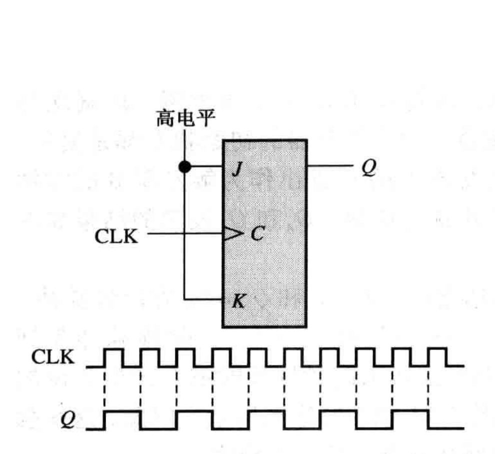


图 7.37　J-K 触发器用做除以 2 芯片，Q 是时钟频率的一半

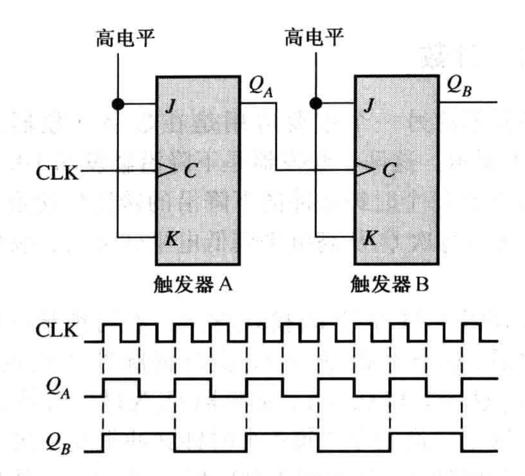


图 7.38　用以把时钟频率除以 4 的两个 J-K 触发器例子。Q_A是时钟频率的二分频，Q_B是时钟频率的四分频

以这种方式连接触发器，就可以实现2^n分频，其中 n 是触发器的个数。例如，3 个触发器把时钟频率除以 $2^3=8$；4 个触发器把时钟频率除以 $2^4=16$；以此类推。

例 7.10　当一个 8 kHz 的方波输入加在触发器 A 的时钟输入时，为图 7.39 中的电路给出 f_{out} 波形。

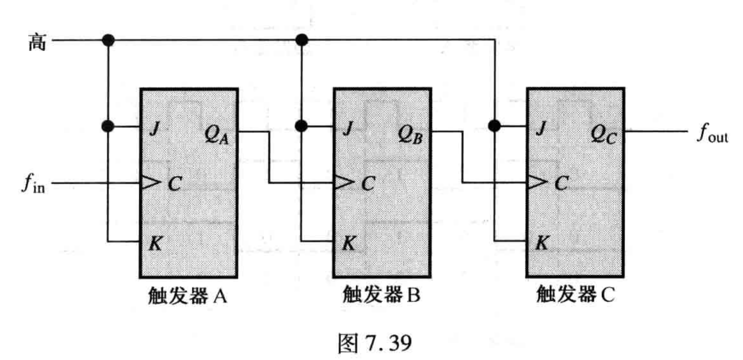


图 7.39

解： 3 个触发器连在一起把输入频率除以 $8(2^3=8)$，f_{out} 的波形如图 7.40 所示。因为这些触发器是上升沿触发，所以输出在上升沿发生变化。每 8 个输入脉冲就有一个输出脉冲，所以输出频率是 1 kHz。Q_A和 Q_B的波形如图所示。

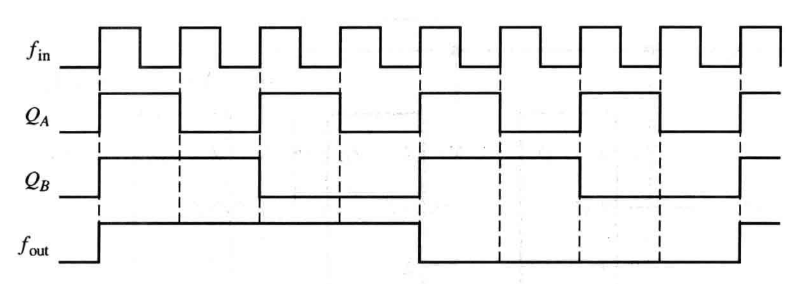


图 7.40

相关问题： 把频率除以 32 需要多少个触发器？

7.4.3 计数

触发器的另一个重要应用是在数字计数器方面，这将在第 8 章详细介绍，其概念如图 7.41 所示。这两个触发器是下降沿触发的 J-K 触发器。两个触发器的初始状态都是复位。触发器 A 在每个时钟脉冲的下降沿的转换处切换。触发器 A 的 Q 输出作为触发器 B 的时钟脉冲，所以每次 Q_A从高电平到低电平转换时，触发器 B 就会切换。Q_A和 Q_B波形的结果如图所示。

观察图 7.41 中的 Q_A和 Q_B序列。在时钟脉冲 1 到达之前，$Q_A=0$ 和 $Q_B=0$；在时钟脉冲 1 到达之后，$Q_A=1$ 和 $Q_B=0$；在时钟脉冲 2 到达之后，$Q_A=0$ 和 $Q_B=1$；在时钟脉冲 3 到达之后，$Q_A=1$ 和 $Q_B=1$。如果取 Q_A为最低有效位，触发器输入时钟脉冲时就会产生 2 位的序列。这个二进制序列每 4 个时钟脉冲重复一次，如图 7.41 中的时序图所示。因此，这些触发器按顺序计数，从 0 到 3(00, 01, 10, 11)，然后返回 0 重新开始这个顺序。

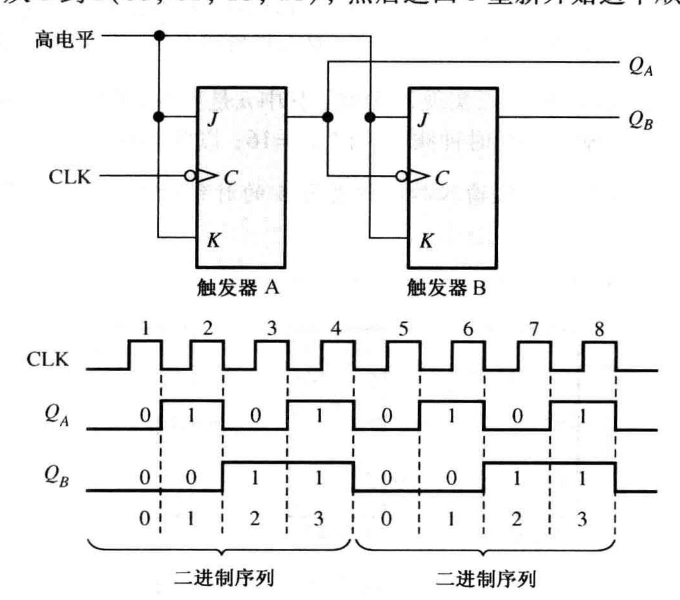


图 7.41　用来生成二进制计数序列的触发器。给出了两个(00, 01, 10, 11)循环

例 7.11　为图 7.42 中电路的 Q_A、Q_B和 Q_C确定与时钟关联的输出波形，并给出由这些波形所表示的二进制序列。

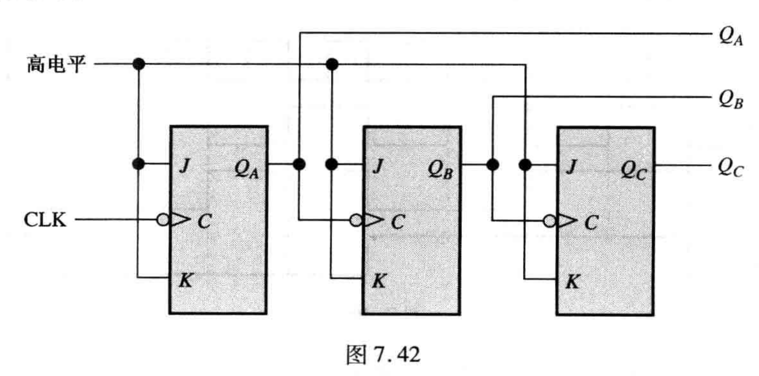


图 7.42

解： 输出时序图如图 7.43 所示。注意输出在时钟脉冲的下降沿改变。输出经过的二进制序列 000，001，010，011，100，101，110，111 如图所示。

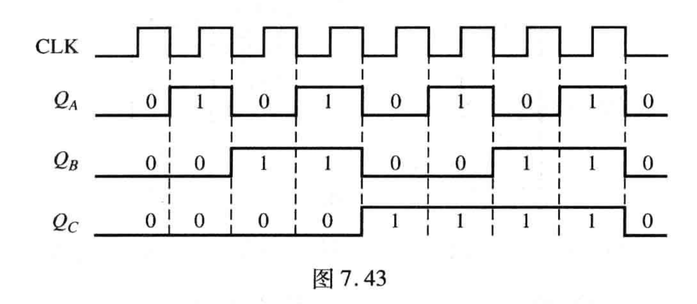


图 7.43

相关问题： 产生表示十进制数 0～15 的二进制序列需要多少个触发器？

7.4 节　温故而知新

1. 用于数据存储的触发器组称为什么？
2. 怎样把 J-K 触发器连接为除以 2 电路？
3. 产生 64 分频的电路需要多少个触发器？

7.5　单稳态触发器

◇ 单稳态触发器每触发一次就产生一个单脉冲。

图 7.44 给出了一个基本的单稳态触发器（单稳态多谐振荡器），它由一个逻辑门和一个反相器组成。当一个脉冲加到触发器的输入时，门 G_1 的输出就变为低电平。这个高电平到低电平的转换通过电容器耦联到反相器 G_2 的输入端。G_2 是在低电平，使得它的输出变为高电平。这个高电平回连到 G_1，使得它的输出保持低电平。此时触发器脉冲已经使得单稳态触发器的输出 Q 变为高电平。

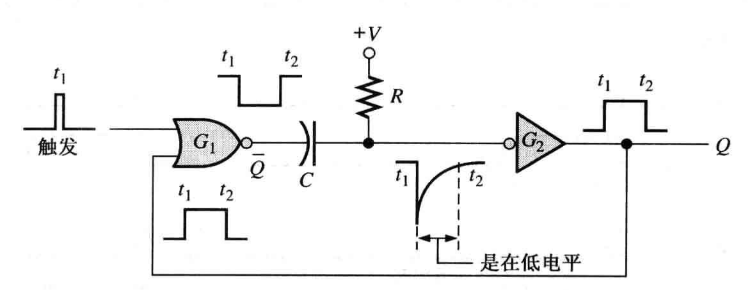


图 7.44　一个简单的单稳态触发器

电容立即开始通过电阻 R 充电直到高电平。充电的速率由 RC 时间常数来确定。当电容充电达到某个电平，这个电平对于 G_2 来说是一个高电平，这时输出就回到低电平。

总结一下，反相器 G_2 的输出响应触发器的输入而变为高电平。它保持高电平的时间由 RC 时间常数设定。在这个时间结束时，输出变为低电平。所以单个窄触发脉冲产生单个输出脉冲，脉冲的持续时间由 RC 时间常数来控制。这个工作过程如图 7.44 所示。

典型的单稳态触发器的逻辑符号如图 7.45(a)所示，而具有外部电阻 R 和电容 C 的相同符号如图 7.45(b)所示。IC 单稳态触发器的两个基本类型是不可重复触发型和可重复触发型。

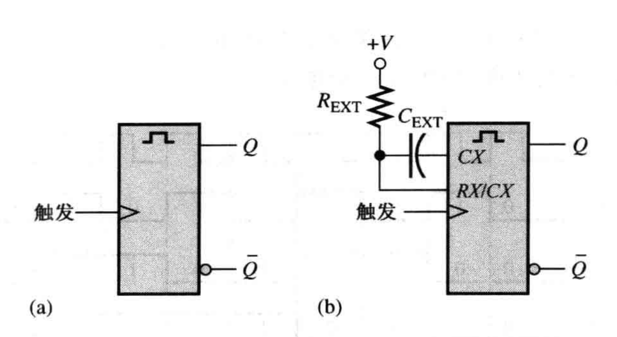


图 7.45 基本单稳态触发器逻辑符号。*CX* 和 *RX* 表示外部元件

不可重复触发单稳态触发器从它触发进入非稳态到返回稳态这段时间，不会响应任何其他触发脉冲。也就是说，在时间结束之前它将忽略任何触发脉冲。单稳态触发器保持为非稳态的时间是输出的脉冲宽度。

图 7.46 给出了不可重复触发单稳态触发器的时序，触发器分别在比它的脉冲宽度大的间期和比它的脉冲宽度小的间期被触发。注意在第二种情况下，附加脉冲被忽略了。

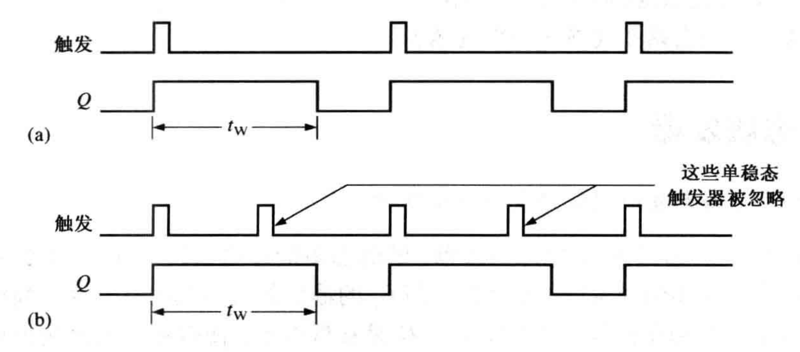


图 7.46 不可重复触发单稳态触发器的响应

可重复触发单稳态触发器可以在它的结束时间之前被触发。重复触发的结果是脉冲宽度的延伸，如图 7.47 所示。

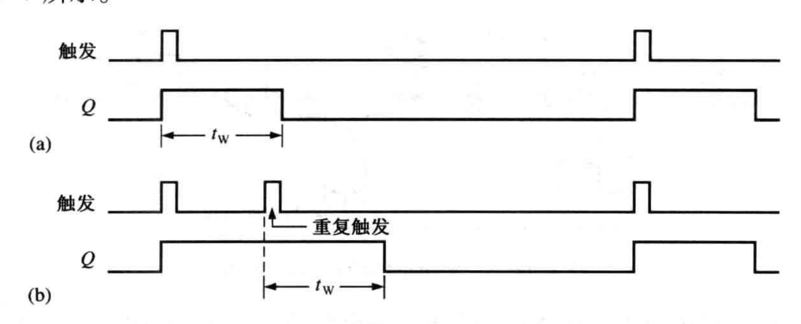
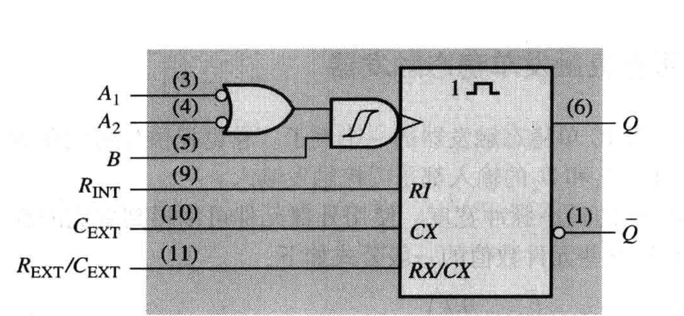


图 7.47 可重复触发单稳态触发器的响应

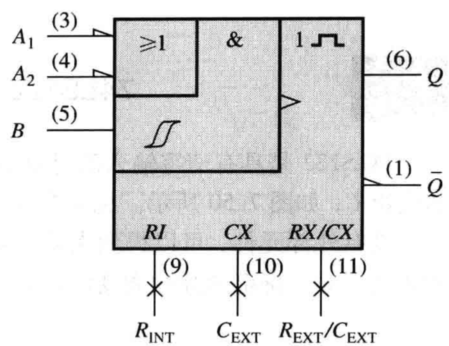


74121 不可重复触发单稳态触发器

74121 是不可重复触发 IC 单稳态触发器的一个例子。芯片提供外部电阻 R 和电容 C，如图 7.48 所示。标记为 A_1、A_2和 B 的输入都是门控触发输入。R_{INT} 输入连接到一个 2 kΩ 的内部定时电阻上。



(a) 传统逻辑符号



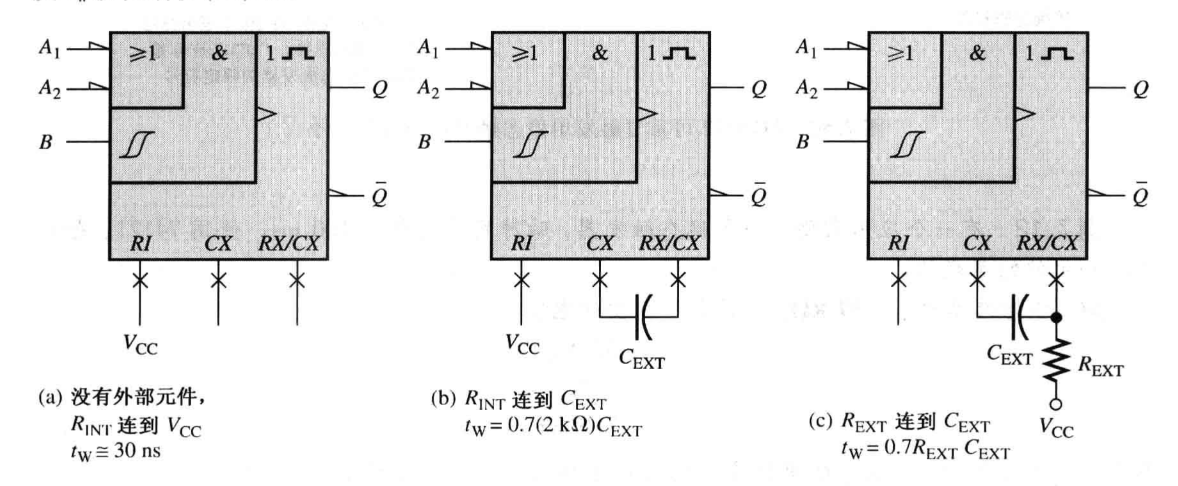
(b) ANSI/IEEE 标准 91-1984 逻辑符号（×＝非逻辑连接），"1⎍"是不可重复触发单稳态触发器的限定符号

图 7.48　74121 不可重复触发单稳态触发器的逻辑符号

设置脉冲宽度　在不使用外部定时元件并且内部定时电阻（R_{INT}）连接到 V_{CC} 时，就会产生一个宽度为 30 ns 的典型脉冲，如图 7.49(a)所示。使用外部元件可以在 30 ns 到 28 s 之间的任何时间间隔内设置脉冲宽度。图 7.49(b)给出了使用内部电阻(2 kΩ)和外部电容的配置。图 7.49(c)给出了使用外部电阻和外部电容的配置。输出脉冲宽度由电阻值（R_{INT} = 2 kΩ 和选择 R_{EXT}）及电容来确定，公式如下所示：

$$t_W = 0.7RC_{EXT} \tag{7.1}$$

其中 R 是 R_{INT} 或者 R_{EXT}。当 R 的单位为千欧姆(kΩ)且 C_{EXT} 的单位为皮法(pF)时，输出脉冲宽度 t_W 就在纳秒(ns)级。



(a) 没有外部元件，R_{INT} 连到 V_{CC}，$t_W \cong 30$ ns

(b) R_{INT} 连到 C_{EXT}，$t_W = 0.7(2\ k\Omega)C_{EXT}$

(c) R_{EXT} 连到 C_{EXT}，$t_W = 0.7R_{EXT}C_{EXT}$

图 7.49　设置 74121 的脉冲宽度的三种方法

施密特触发符号　符号⎰⎱表示施密特输入。这种类型的输入使用产生磁滞现象的特殊阈值电路，这种特征可以在缓慢变化的触发电压徘徊在临界输入电平周围时，防止状态之间的不稳定翻转。即使当输入的变化速度缓慢到 1 V/s 时，也可以发生可靠的触发。



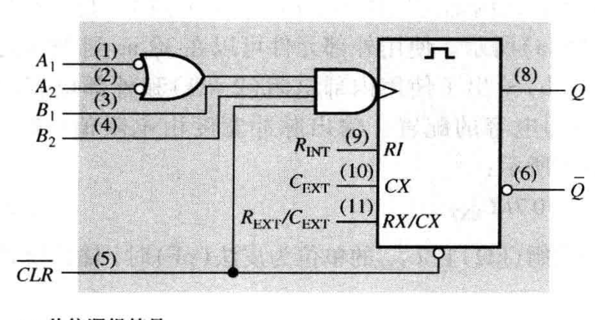
74LS122 可重复触发单稳态触发器

74LS122 是具有清零输入的可重复触发 IC 单稳态触发器的一个例子。它也提供外部电阻 R 和电容 C，如图 7.50 所示。标记为 A_1、A_2、B_1 和 B_2 的输入都是门控触发输入。

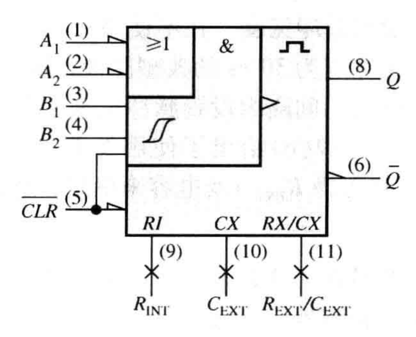
没有外部元件，可以得到大约为 45 ns 的最小脉冲宽度。使用外部元件可以得到更宽的脉冲宽度。对于特定的脉冲宽度(t_W)，计算这些元件数值的一般公式如下：

$$t_W = 0.32RC_{EXT}\left(1 + \frac{0.7}{R}\right) \tag{7.2}$$

其中 0.32 是由特定类型的单稳态触发器确定的一个常数，R 为 kΩ 级，它可以是内部也可以是外部电阻，C_{EXT} 的单位为 pF，而 t_W 的单位为 ns。内部电阻是 10 kΩ，可以用外部电阻替代。[注意此式和 74121 的式(7.1)之间的区别。]



(a) 传统逻辑符号



(b) ANSI/IEEE 标准 91-1984 逻辑符号(×=非逻辑连接)，"⎍"是可重复触发单稳态触发器的限定符号

图 7.50 74LS122 可重复触发单稳态触发器的逻辑符号

例 7.12 有一个应用需要一个单稳态触发器，脉冲宽度大约为 100 ms。使用 74121，给出连接和元件的参数值。

解： 任意选择 R_{EXT} = **39 kΩ**，并计算所需要的电容。

$$t_W = 0.7R_{EXT}C_{EXT}$$

$$C_{EXT} = \frac{t_W}{0.7R_{EXT}}$$

其中 C_{EXT} 的单位为 pF，R_{EXT} 的单位为 kΩ，t_W 的单位为 ns。由于 100 ms = 1×10^8 ns，

$$C_{EXT} = \frac{1 \times 10^8\ \text{ns}}{0.7(39\ \text{k}\Omega)} = 3.66 \times 10^{-6}\ \text{pF} = \mathbf{3.66\ \mu F}$$

标准 3.3 μF 电容将给出一个宽度为 91 ms 的输出脉冲。正确的连接如图 7.51 所示。为了获得接近 100 ms 的脉冲宽度，可以尝试 R_{EXT} 和 C_{EXT} 的其他数值组合。例如，R_{EXT} = 68 kΩ 和 C_{EXT} = 2.2 μF 可以给出的脉冲宽度为 105 ms。

相关问题： 使用外部电容并和 R_{INT} 连接，利用 74121 产生宽度为 10 μs 的输出脉冲。

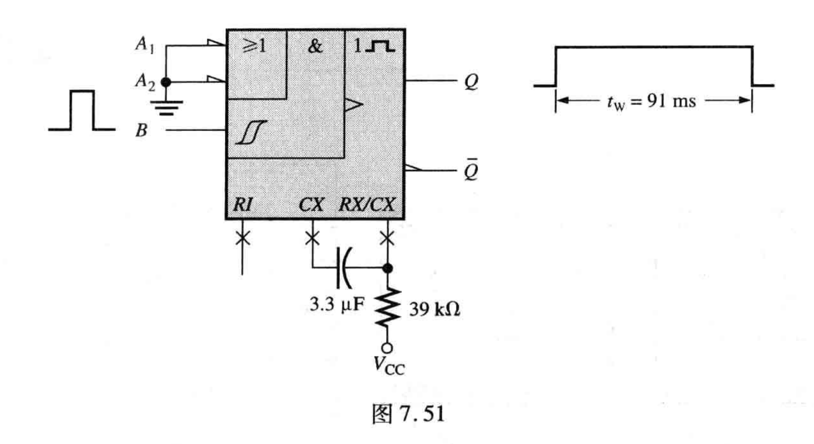


图7.51

例7.13 使用74LS122连接R_{EXT}和C_{EXT}，产生一个宽度为1 μs的输出脉冲，确定R_{EXT}和C_{EXT}的数值。

解： 假设C_{EXT} = **560 pF**，然后解出R_{EXT}。脉冲宽度必须表示为ns，C_{EXT}必须表示为pF，R_{EXT}将是kΩ。

$$\begin{aligned}
t_W &= 0.32R_{EXT}C_{EXT}\left(1+\frac{0.7}{R_{EXT}}\right) = 0.32R_{EXT}C_{EXT} + 0.7\left(\frac{0.32R_{EXT}C_{EXT}}{R_{EXT}}\right) \\
&= 0.32R_{EXT}C_{EXT} + (0.7)(0.32)C_{EXT} \\
R_{EXT} &= \frac{t_W-(0.7)(0.32)C_{EXT}}{0.32C_{EXT}} = \frac{t_W}{0.32C_{EXT}} - 0.7 \\
&= \frac{1000\ \text{ns}}{(0.32)560\ \text{pF}} - 0.7 = \mathbf{4.88\ k\Omega}
\end{aligned}$$

使用4.7 kΩ的标准数值。

相关问题： 对于74LS122单稳态触发器，具有宽度为5 μs的输出脉冲，给出连接和元件的参数值。假设C_{EXT} = 560 pF。

7.5.1 应用举例

一个实际的单稳态触发器应用是序列定时器，可以用来点亮一系列电灯。例如，这种类型的电路可以在高速公路建设项目中用于车道改变的定向指示器，或者用于车辆的系列信号灯的打开。

图7.52给出了3个74LS122单稳态触发器连接在一起用做序列定时器。这个特殊电路产生3个1秒脉冲的序列。第一个单稳态触发器由开关关闭或者低频脉冲输入来触发，产生一个1秒输出脉冲。当第一个单稳态触发器(OS 1)定时结束时，1秒脉冲就变为低电平，第二个单稳态触发器(OS 2)随即被触发，同样产生一个1秒输出脉冲。当第二个脉冲变为低电平时，第三个单稳态触发器(OS 3)就会被触发，并且产生第三个1秒脉冲。输出时序图如图7.52所示。这个基本安排的改变可以用来产生不同的定时输出。

7.5.2 555定时器

555定时器是功能多样和广泛应用的IC芯片，因为它可以用两种不同的模式来配置，要么是单稳态多谐振荡器(单稳态触发器)，要么是非稳态多谐振荡器(振荡器)。非稳态多谐振荡器将在7.6节讨论。

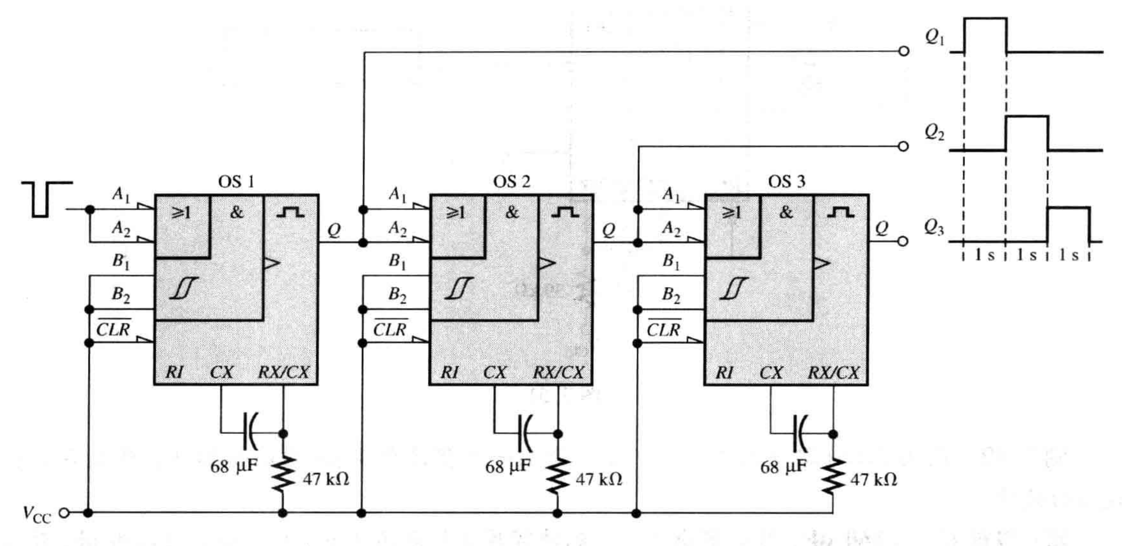


图 7.52　使用 3 个 74LS122 单稳态触发器的序列定时电路

555 定时器的工作　如图 7.53 所示给出 555 定时器内部元件的功能图。比较器芯片的特点是，当其正(+)输入上的电压大于负(-)输入上的电压时，输出就是高电平，而当负输入电压大于正输入电压时，输出就是低电平。分压器由三个 5 kΩ 电阻组成，提供了 $1/3V_{CC}$的触发电平及 $2/3V_{CC}$的阈值电平。控制电压输入(引脚 5)可以根据需要，用来从外部把触发和阈值电平改变为其他的值。当通常情况下的高电平触发输入暂时低于 $1/3V_{CC}$时，比较器 B 的输出就从低电平变为高电平，并且使 S-R 锁存器置位，使得输出(引脚 3)变为高电平，并且使放电晶体管 Q_1 截止。输出将保持在高电平直至正常的低电平阈值输入高于$2/3V_{CC}$，并且使得比较器 A 的输出从低电平变为高电平。这时锁存器复位，使得输出回到低电平并且使放电晶体管导通。外部的复位输入可以用来复位锁存器，此锁存器独立于阈值电路。触发和阈值输入(引脚 2 和引脚 6)由外部连接的元件来控制，从而产生单稳态或者非稳态的效果。

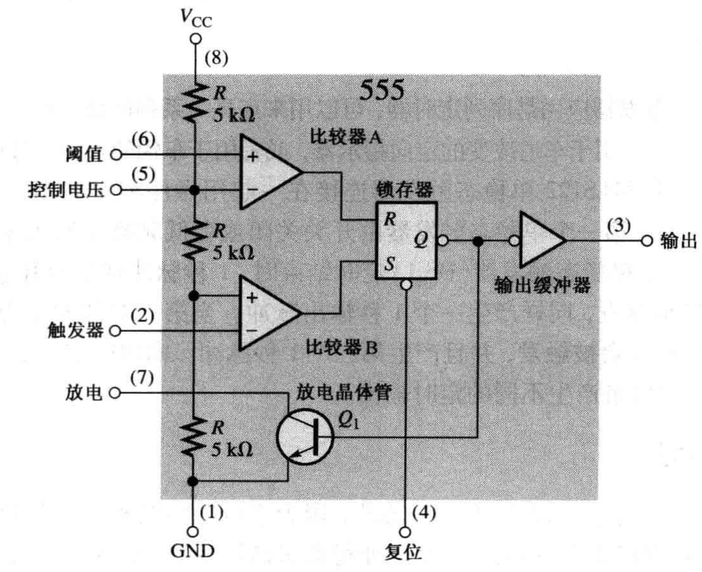


图 7.53　555 定时器(引脚编号在括号内)的内部功能图

单稳态(单稳态触发器)的工作　外部电阻和电容的连接如图7.54所示，用以设置555定时器为不可重复触发的单稳态触发器。输出的脉冲宽度由时间常数 R_1 和 C_1 根据下面的公式来确定：

$$t_W = 1.1R_1C_1 \tag{7.3}$$

没有使用控制电压输入，而是连接一个去偶电容器 C_2 来防止噪声影响触发和阈值电平。

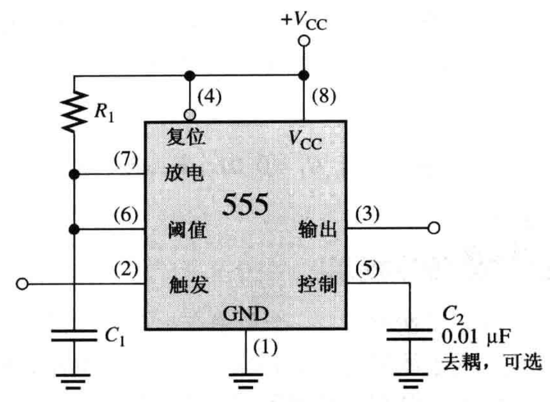
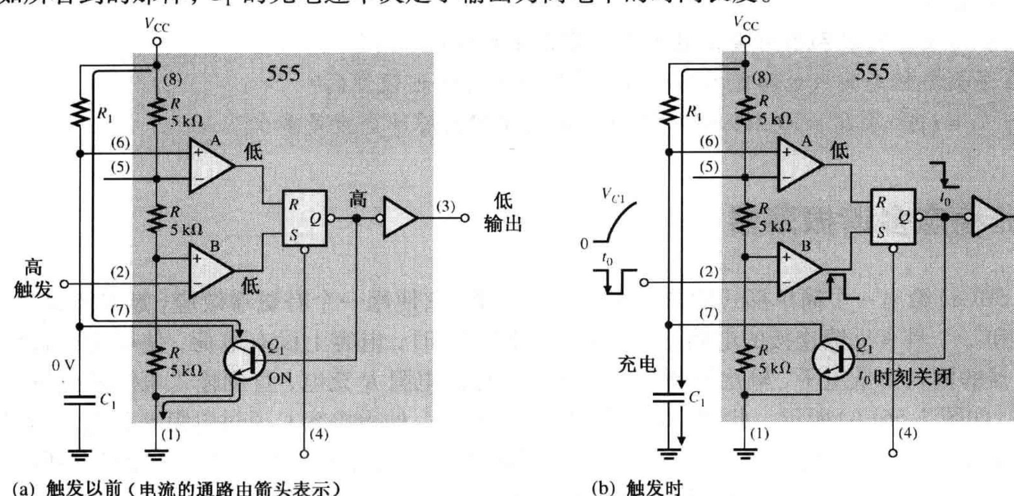
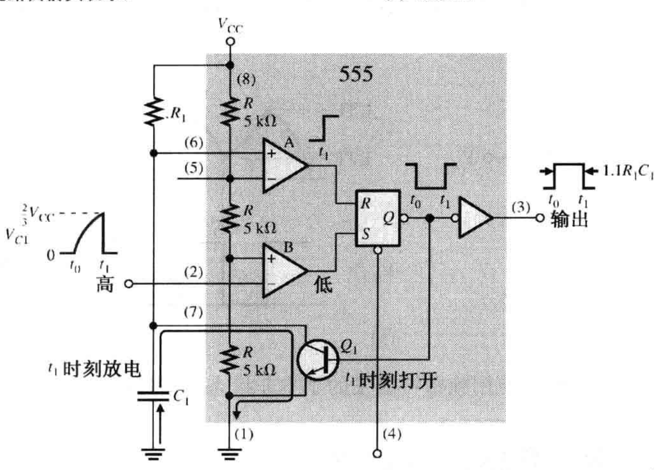


图 7.54　555 定时器连接成单稳态触发器

在施加触发脉冲之前，输出为低电平，放电晶体管 Q_1 导通，保持 C_1 放电，如图7.55(a)所示。在 t_0 时加上下降沿触发脉冲，输出变为高电平而使得放电晶体管截止，允许电容器 C_1 开始通过 R_1 充电，如图7.55(b)所示。当 C_1 充电到1/3V_{CC}时，输出将在 t_1 时刻回到低电平，而 Q_1 立即导通，使 C_1 放电，如图7.55(c)所示。正如所看到的那样，C_1 的充电速率决定了输出为高电平的时间长度。



(a) 触发以前(电流的通路由箭头表示)　(b) 触发时



(c) 充电期间快结束时

图 7.55　555 定时器的单稳态触发器运算

例 7.14 $R_1 = 2.2\ \text{k}\Omega$ 且 $C_1 = 0.01\ \mu\text{F}$ 的 555 单稳态电路的输出脉冲宽度为多少?

解: 从式(7.3)得到脉冲宽度为

$$t_W = 1.1R_1C_1 = 1.1(2.2\ \text{k}\Omega)(0.01\ \mu\text{F}) = \mathbf{24.2\ \mu s}$$

相关问题: 对于 $C_1 = 0.01\ \mu\text{F}$,确定脉冲宽度为 1 ms 的 R_1 的数值。



相应提示

在正常情况下,一个单稳产生单一的脉冲,这很难在示波器上测量,因为脉冲的产生不规则。为了测量的目的获取一个稳定的显示,从一个脉冲生成器产生一个周期比预期脉冲宽度长的同样脉冲来触发示波器的方法很有用。对于比较长的脉冲,使用数字示波器存储波形或通过一些已知的参数来减小时间常数。例如,用 1000 μF 的电容替代 1 μF 的电容以减小时间 1000 倍。一个较快的脉冲更容易在示波器上观察和测量。

7.5 节 温故而知新

1. 描述可重复触发和不可重复触发的单稳态触发器的不同之处。
2. 在多数单稳态集成电路芯片中,输出脉冲宽度是如何设置的?
3. 当 $C = 1\mu\text{F}$ 且 $R = 10\ \text{k}\Omega$ 时,555 定时器的单稳态脉冲宽度是多少?

7.6 非稳态多谐振荡器

图 7.56(a)给出一个简单形式的非稳态多谐振荡器,它使用一个有磁滞效应(施密特触发)的反相器和一个具有反馈连接的电路。当电源第一次加上时,电容上没有电荷,使得施密特触发的反相器的输入为低电平,输出为高电平。电容 C 通过电阻 R 充电,直到输入电压达到触发点(UTP),如图 7.56(b)所示。此时,反相器输出为高电平,使得电容 C 通过电阻 R 放电。当反相器输入电压降低到低于触发点(LTP)时,输出变为高电平,电容 C 再次充电。只要没有断电,这个充电/放电循环过程一直持续重复,输出就是脉冲波,如图中给出的那样。

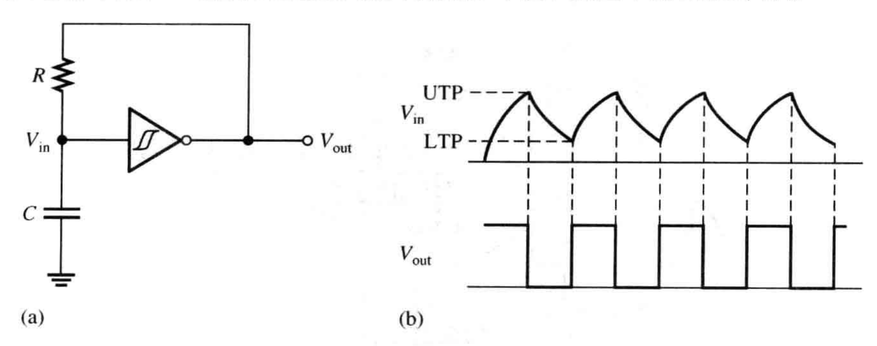


图 7.56 使用施密特触发的基本非稳态多谐振荡器

7.6.1 555 定时器用做非稳态多谐振荡器

555 定时器连接后作为非稳态多谐振荡器使用,如图 7.57 所示。注意阈值输入现在连接到

了触发输入上。外部元件 R_1、R_2和 C_1构成了定时网络，用以设置振荡器的频率。0.01 μF 电容器 C_2连接于控制输入；C_2严格用于去耦，对运算没有影响，在某些情况下可以去掉。

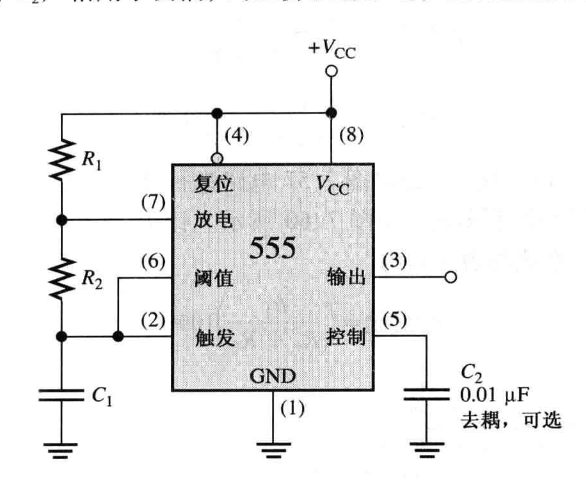


图 7.57　连接为非稳态多谐振荡器(振荡器)的 555 定时器



计算机小知识

所有的计算机都需要一个定时源来提供精确的时钟脉冲。定时部分控制所有的系统时序，并且负责系统硬件的正确运算。定时部分通常由晶体控制的振荡器和用以分频的计数器组成。使用高频振荡器分频为低频，可以得到更准确和稳定的频率。

开始时，当电源打开后，电容器 (C_1) 并没有充电，因此触发电压(引脚 2)是 0 V。这导致了比较器 B 的输出为高电平，比较器 A 的输出为低电平，迫使锁存器的输出，也就是基极 Q_1 为低电平，并保持晶体管截止。这时，C_1 开始通过 R_1 和 R_2充电，如图 7.58 所示。当电容器电压达到 $1/3V_{CC}$时，比较器 B 就变为它的低电平输出状态；而当电容器电压达到 $2/3V_{CC}$时，比较器 A 就变为它的高电平输出状态。这就使锁存器复位，使得基极 Q_1 变为高电平，并使晶体管导通。这一连串变化为电容器建立了一个从 R_2到晶体管的放电回路，如图所示。电容器就在开始放电，使得比较器 A 变为低电平。当电容器放电到 $1/3V_{CC}$时，比较器 B 就变为高电平；这使得锁存器置位，并使得基极 Q_1 变为低电平，并且关闭晶体管。然后开始另一个充电循环，重复整个过程。结果就是一个矩形波输出，它的占空比取决于电阻 R_1和 R_2的数值。振荡的频率由下面的公式给出，或者可以在图 7.59 中找到。

$$f = \frac{1.44}{(R_1 + 2R_2)C_1}$$

通过选择 R_1和 R_2，可以调节输出的占空比。由于 C_1通过 $R_1 + R_2$充电，仅通过 R_2放电，如果 $R_2 \gg R_1$，就可以达到最接近 50% 的占空比，从而使得充电和放电时间大致相等。

占空比的表达式如下所示。输出为高电平的时间(t_H)是 C_1 从 $1/3V_{CC}$充电到 $2/3V_{CC}$所需要的时间。它可以表示为

$$t_H = 0.7(R_1 + R_2)C_1 \tag{7.5}$$

输出为低电平的时间(t_L)是 C_1 从 $1/3V_{CC}$放电到 $2/3V_{CC}$所需要的时间。它可以表示为

$$t_L = 0.7R_2C_1 \tag{7.6}$$

输出波形的周期 T 是 t_H和 t_L的和。这就是式(7.4)中 f 的倒数。

$$T = t_H + t_L = 0.7(R_1 + 2R_2)C_1$$

最后，占空比为

$$占空比 = \frac{t_H}{T} = \frac{t_H}{t_H + t_L}$$

$$占空比 = \left(\frac{R_1 + R_2}{R_1 + 2R_2}\right)100\% \tag{7.7}$$

为了实现小于50%的占空比，可以对图7.57电路进行修改，使得C_1通过R_1充电，并通过R_2放电。这由一个二极管D_1来实现，如图7.60所示。让R_1小于R_2就可以做到占空比小于50%。在这种情况下，占空比的表达式为

$$占空比 = \left(\frac{R_1}{R_1 + R_2}\right)100\% \tag{7.8}$$

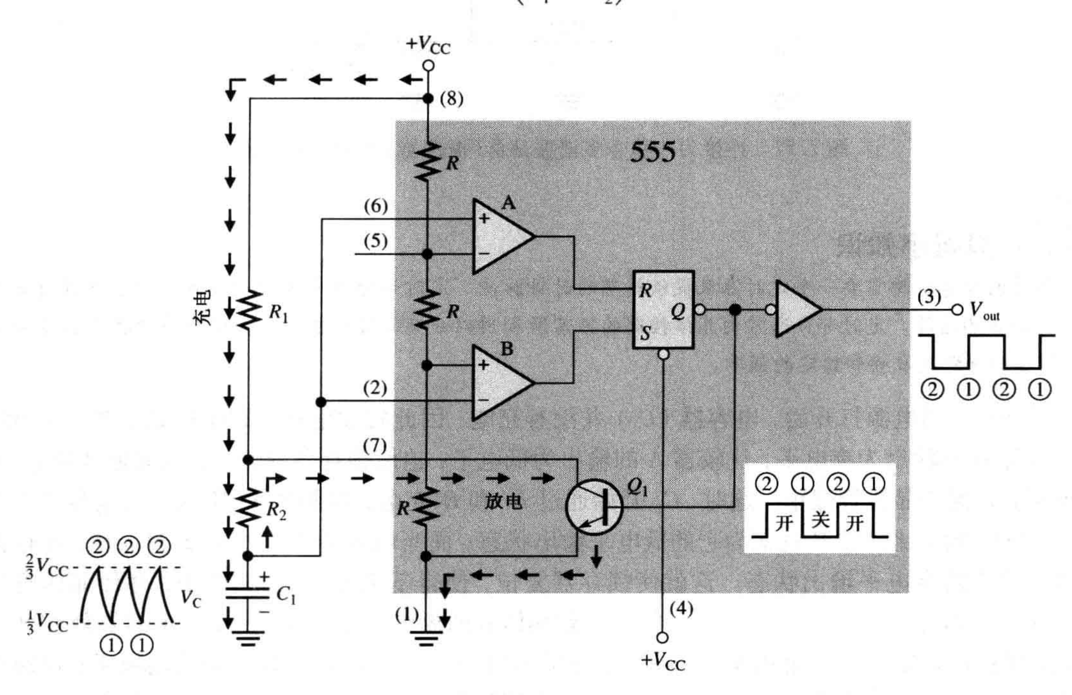


图7.58　在非稳态模式下的555定时器的运行

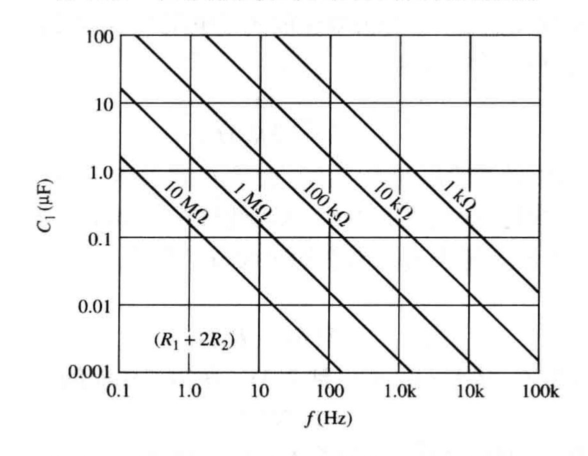


图7.59　作为C_1和R_1+2R_2函数的振荡频率。斜线是R_1+2R_2的值

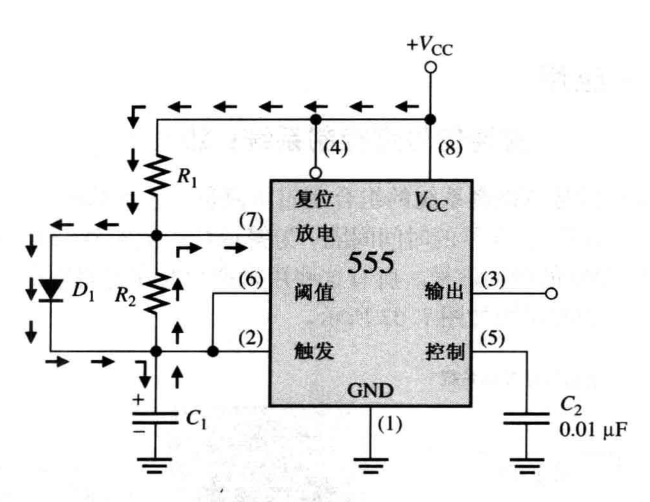


图 7.60 添加二极管 D_1，使 $R_1 < R_2$，这样就允许输出的占空比调节到小于 50%

例 7.15 555 定时器设置运行于非稳态模式（振荡器），如图 7.61 所示。确定输出频率和占空比。

解：使用式(7.4)和式(7.7)，

$$f = \frac{1.44}{(R_1 + 2R_2)C_1} = \frac{1.44}{(2.2\ \mathrm{k\Omega} + 9.4\ \mathrm{k\Omega})0.022\ \mu\mathrm{F}} = \mathbf{5.64\ kHz}$$

$$\text{占空比} = \left(\frac{R_1 + R_2}{R_1 + 2R_2}\right)100\% = \left(\frac{2.2\ \mathrm{k\Omega} + 4.7\ \mathrm{k\Omega}}{2.2\ \mathrm{k\Omega} + 9.4\ \mathrm{k\Omega}}\right)100\% = \mathbf{59.5\%}$$

相关问题：如图 7.60 所示，在 R_2 两端跨接一个二极管，确定图 7.61 中的占空比。

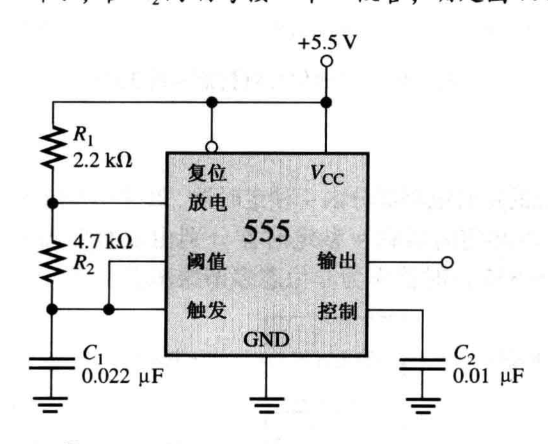


图 7.61 打开文件 F07-61 检验该操作

7.6 节 温故而知新

1. 解释非稳态多谐振荡器和单稳态触发器的不同之处。
2. 某个非稳态多谐振荡器的 $t_H = 15$ ms 和 $T = 20$ ms，输出的占空比是多少？



数字系统应用

交通信号灯控制系统：部分 2

第 6 章介绍了交通信号灯控制系统的组合逻辑电路部分，本章将介绍定时电路。回顾这些电路，它为红灯和绿灯产生 25 秒的时间间隔，为警示灯产生 4 秒的时间间隔。同时，通过定时电路产生一个 10 kHz 的时钟信号。所有这些用来产生时序逻辑输出，将在第 8 章进行设计。全部的交通信号灯控制系统如图 7.62 所示。

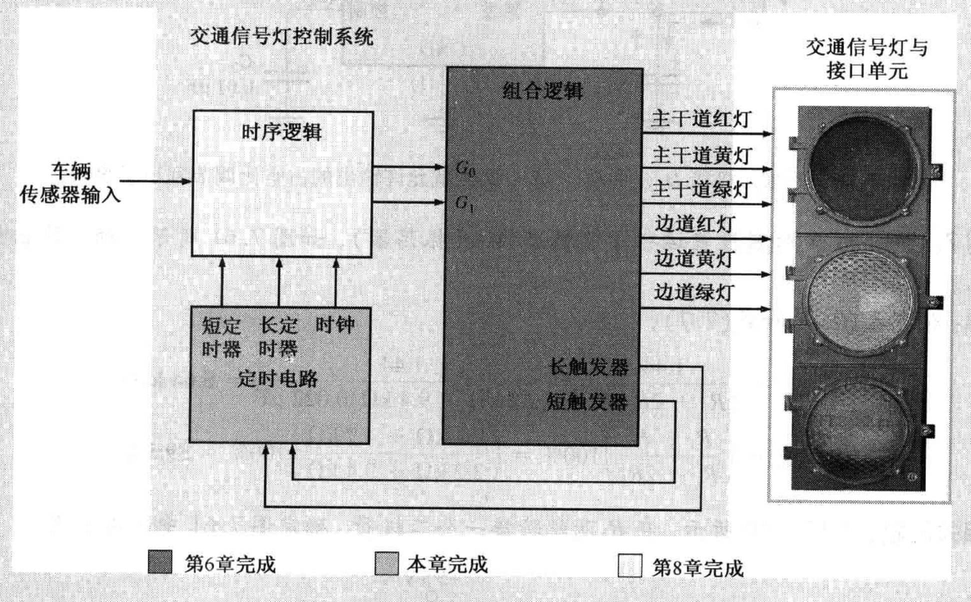


图 7.62　交通信号灯控制系统框图

定时电路

交通信号灯控制系统的定时电路部分由 4 秒定时器、25 秒定时器和 10 kHz 振荡器组成，如图 7.63 中的框图所示。25 秒定时器和 4 秒定时器分别由长触发器和短触发器触发，形成组合逻辑。10 kHz振荡器由 555 定时器作为非稳态多谐振荡器实现。

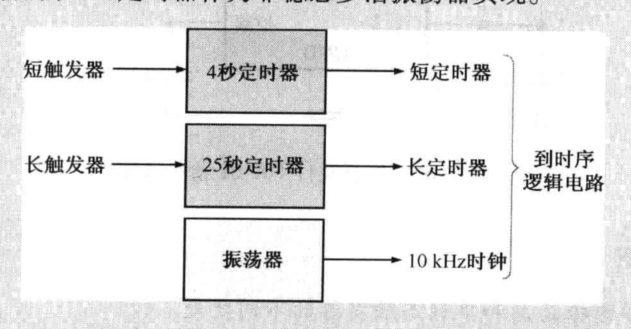


图 7.63　定时电路的框图

25 秒定时器和 4 秒定时器由 555 定时器配置成单稳态(单稳)来完成，如图 7.64(a)所示。图 7.64(b)为 10 kHz 振荡器。作为替换，74121 单稳可以用来实现 25 秒定时器和 4 秒定时器。

1. 为25秒定时器确定外部电阻 R 和电容 C 的值。
2. 为4秒定时器确定外部电阻 R 和电容 C 的值。
3. 为10 kHz振荡器确定两个电阻和电容的值。
4. 使用74121单稳重新设计25秒定时器和4秒定时器。

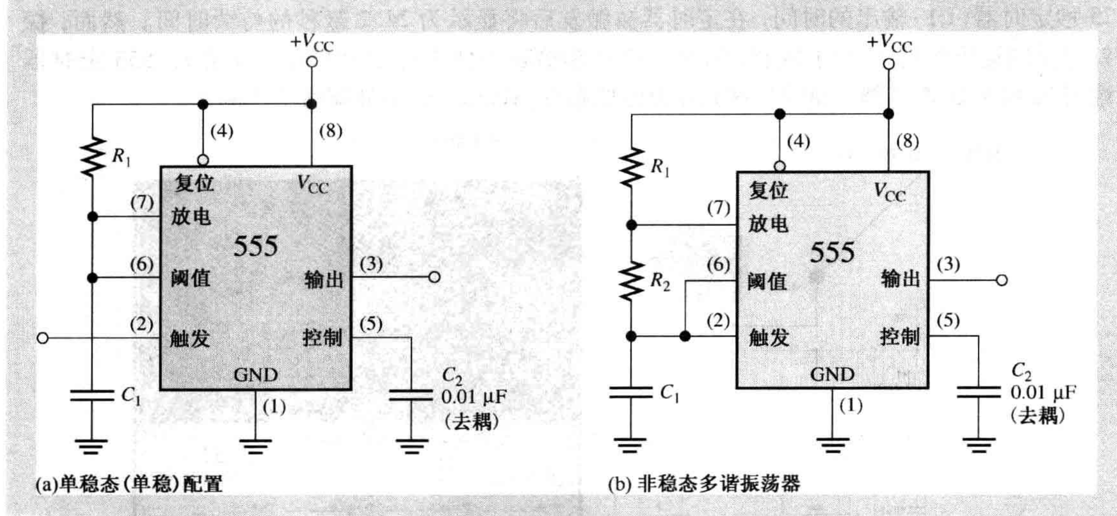


图7.64 定时电路

计算机仿真

图7.65给出了交通信号灯控制系统部分时序电路的计算机仿真的仿真屏幕界面，包括25秒定时器、4秒定时器和10 kHz振荡器。

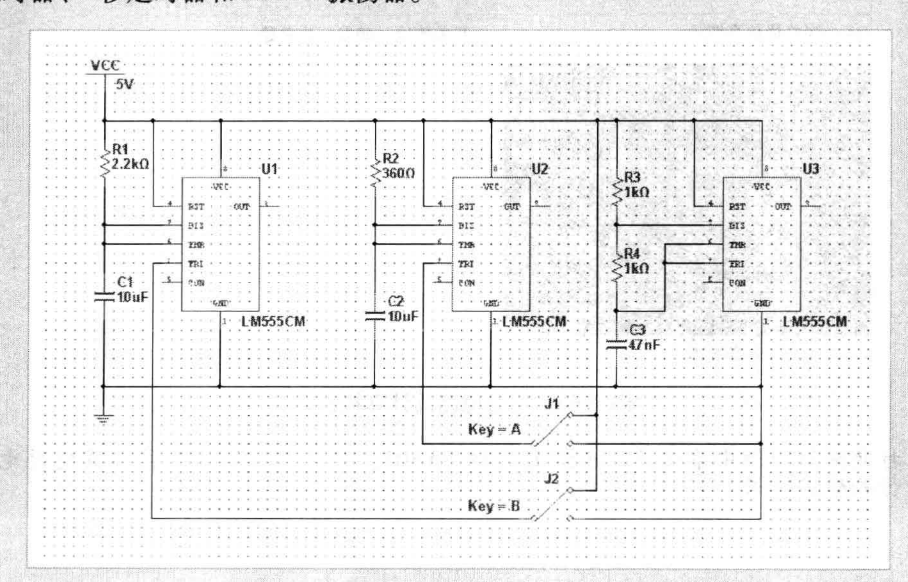
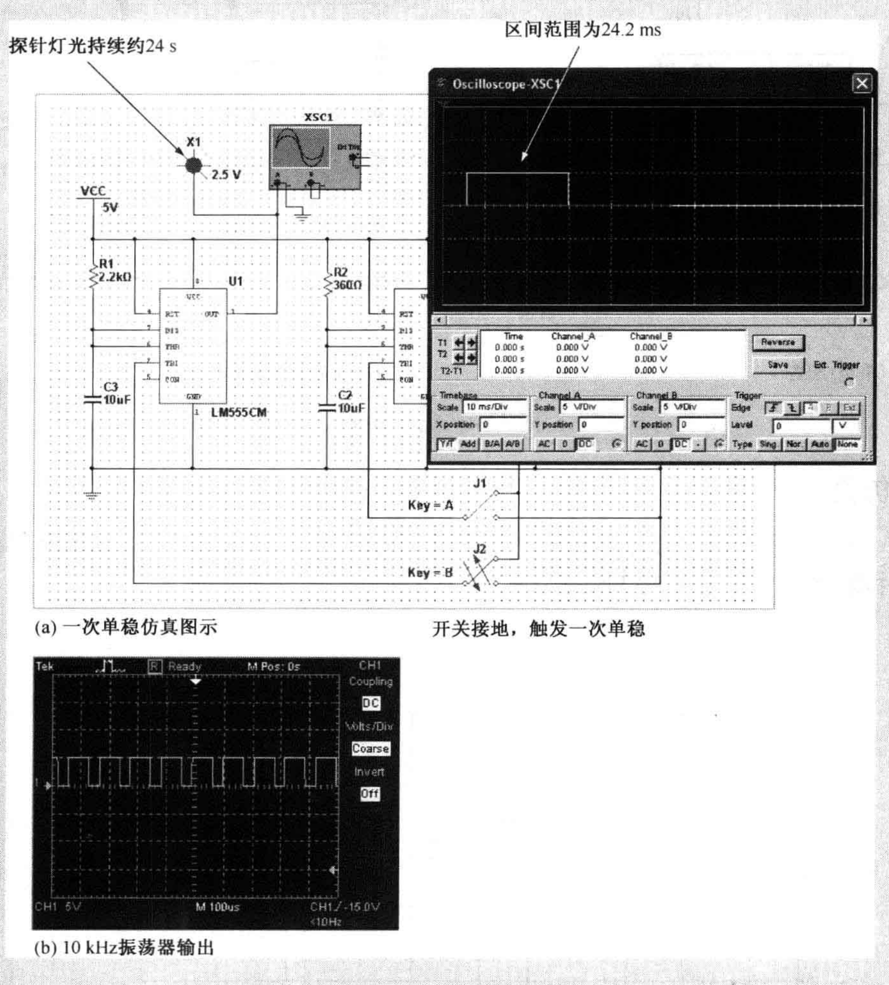


图7.65 时序电路的Multisim输出。开关仅用于测试目的

对应于实际时间的仿真时间 基于软件程序和计算机的仿真时间，要比实际电路里的运行时间快得多。图7.65给出了用于产生仿真的特定的计算机，仿真时间和实际时间的比为1000，

但是依据运行软件的计算机的速度不同而不同，含有一些其他的因素。用于实际时间和仿真时间的这个比，单稳的 *RC* 时间常数值要小于实际使用的硬件电路的值的千分之一。通过将 Multisim 探针连接到一个单稳，并且使用开关触发单稳，就可以观察实时的输出。一台连接到输出的虚拟示波器记录的时间比实时观察到的时间短得多。例如，如图 7.66(a)中连接到 25 秒定时器(U1)输出的时间，在定时器被触发后将显示为 24.2 毫秒的持续时间。然而，探针连接的输出将持续大约 24 秒(依据计算机和仿真中使用的定时步骤的次数)。555 定时器配置为 10 kHz 振荡器，如图 7.66(b)的虚拟泰克(Tektronix)示波器所给出的。



(a) 一次单稳仿真图示

(b) 10 kHz振荡器输出

图 7.66　定时电路的计算机仿真

打开网络资源中系统应用示例的文件 SAA07。使用 Multisim 软件仿真时序电路，运行并观察结果。

关键词

非稳态　没有固定的状态。一个非稳态多谐振荡器在两种半稳定状态。

双稳态　有两种稳定状态。触发器和锁存器是双稳态器件。

清零　使用一个异步输入把一个触发器复位(使得 Q 输出为0)。

时钟　一个触发器的触发输入。

D触发器　一种双稳态器件，设定为在时钟边沿的作用下，D触发器的输入跟随输出。

边沿触发的触发器　一种触发器，在时钟边沿的作用下，输入数据出现在输出上。

保持时间　为了可靠地运行，在时钟的触发边沿到来之后，维持触发器输入的控制电平所需要的时间间隔。

J-K触发器　一种触发器的类型，它可以处于置位、复位、保持和切换模式下。

锁存器　一种双稳态数字电路，用于存储一个位。

单稳　仅有一个稳定状态。单稳多谐振荡器通常称为单稳。

功率损耗　一个电路需要的能量的计量。

预置位　一个异步输入，用于将一个触发器置位(使得 Q 输出为1)。

传输延迟时间　加入输入信号并导致输出发生变化所需的时间间隔。

复位　触发器或锁存器输出为0的一种状态；产生复位状态的一个作用。

置位　触发器或锁存器输出为1的一种状态；产生置位状态的一个作用。

建立时间　在先于时钟触发边沿到来之前，输入控制信号作用在一个数字电路(如一个触发器)输入上所需要的持续时间。

同步　有一个固定的时间关系。

定时器　一种电路，可以用做单稳或振荡器。

切换　一个触发器的工作状态，在每一个时钟脉冲作用下状态都会改变。

判断题　(答案在本章的结尾。)

1. 一个锁存器有两种稳定的状态。
2. 当 Q 输出为低电平时，锁存器被认为处于置位状态。
3. 为了改变状态，门控D锁存器必须使能。
4. 触发器和锁存器都是双稳芯片。
5. 无论 D 输入在什么时间变化，边沿触发的D触发器都会改变状态。
6. 对于一个边沿触发的触发器，时钟输入是必须的。
7. 当 J 和 K 输入为高电平时，一个边沿触发的J-K触发器在每个时钟脉冲作用下都改变状态。
8. 一个单稳被看做一个非稳态多谐振荡器。
9. 在触发作用下，一个单稳输出一个单一的脉冲。
10. 555定时器可以用做单稳或脉冲振荡器。

自测题　(答案在本章的结尾。)

1. 如果S-R锁存器在 S 输入有一个1，在 R 输入上有一个0，然后 S 输入变为0，那么锁存器将会
   (a)置位　(b)复位　(c)无效　(d)清零
2. 当出现下列哪种条件时，S-R锁存器就会出现无效状态？
   (a) $S=1$, $R=0$　(b) $S=0$, $R=1$　(c) $S=1$, $R=1$　(d) $S=0$, $R=0$
3. 对于门控D锁存器，哪种情况下 Q 输出总是等于 D 输入？
   (a)在启动脉冲之前　(b)在启动脉冲期间　(c)在启动脉冲之后　(d)答案(b)和(c)
4. 与锁存器相似，触发器所属的逻辑电路类别称为
   (a)单稳态多谐振荡器　(b)双稳态多谐振荡器

(c)非稳态多谐振荡器　　(d)单次振荡器

5. 触发器的时钟输入的目的是
   (a)清除芯片
   (b)设置芯片
   (c)总是使得输出改变状态
   (d)使得输出呈现的状态取决于控制(S-R、J-K 或者 D)输入
6. 对于边沿触发的 D 触发器，
   (a)触发器状态的改变只发生在时钟脉冲边沿　　(b)触发器要进入的状态取决于 D 输入
   (c)输出跟随每一个时钟脉冲的输入　　(d)所有这些答案
7. 区分 J-K 触发器和 S-R 触发器特征的是
   (a)切换情况　　(b)预置位输入　　(c)时钟类型　　(d)清零输入
8. 触发器处于切换情况，当
   (a)$J=1$，$K=0$　　(b)$J=1$，$K=1$　　(c)$J=0$，$K=0$　　(d)$J=0$，$K=1$
9. J-K 触发器的输入 $J=1$ 和 $K=1$，时钟输入频率为 10 kHz。Q 输出为
   (a)保持为高电平　　(b)保持为低电平　　(c) 10 kHz 方波　　(d)5 kHz 方波
10. 单稳态触发器是下面哪种类型？
    (a)单稳态多谐振荡器　　(b)非稳态多谐振荡器
    (c)定时器　　(d)答案(a)和(c)
    (e)答案(b)和(c)
11. 非可重复触发单稳态触发器的输出脉冲宽度取决于
    (a)触发时间间隔　　(b)电源电压　　(c)电阻和电容　　(d)阈值电压
12. 非稳态多谐振荡器
    (a)需要周期触发输入　　(b)没有稳定状态
    (c)是一个振荡器　　(d)产生周期脉冲输出
    (e)答案(a)、(b)、(c) 和(d)　　(f)只是答案(b)、(c)和(d)

习题

7.1 节　锁存器

1. 如果图 7.67 中的波形应用于低电平有效输入 S-R 锁存器，绘制出和输入相关的 Q 输出波形。假设 Q 开始于低电平。

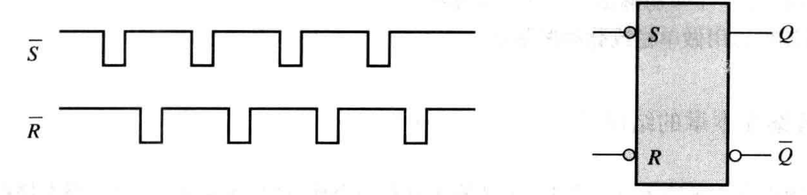


图 7.67

2. 在图 7.68 中的输入波形加到低电平有效的 S-R 锁存器上，给出如习题 1 的解。
3. 对于图 7.69 的输入波形，给出如习题 1 的解。
4. 门控 S-R 锁存器的输入如图 7.70 所示，确定输出 Q 和 $\overline{Q}$ 。给出它们和使能输入的正确关系。假设 Q 开始时是低电平。
5. 对于图 7.71 输入，求如习题 4 的解。
6. 对于图 7.72 的输入，求如习题 4 的解。

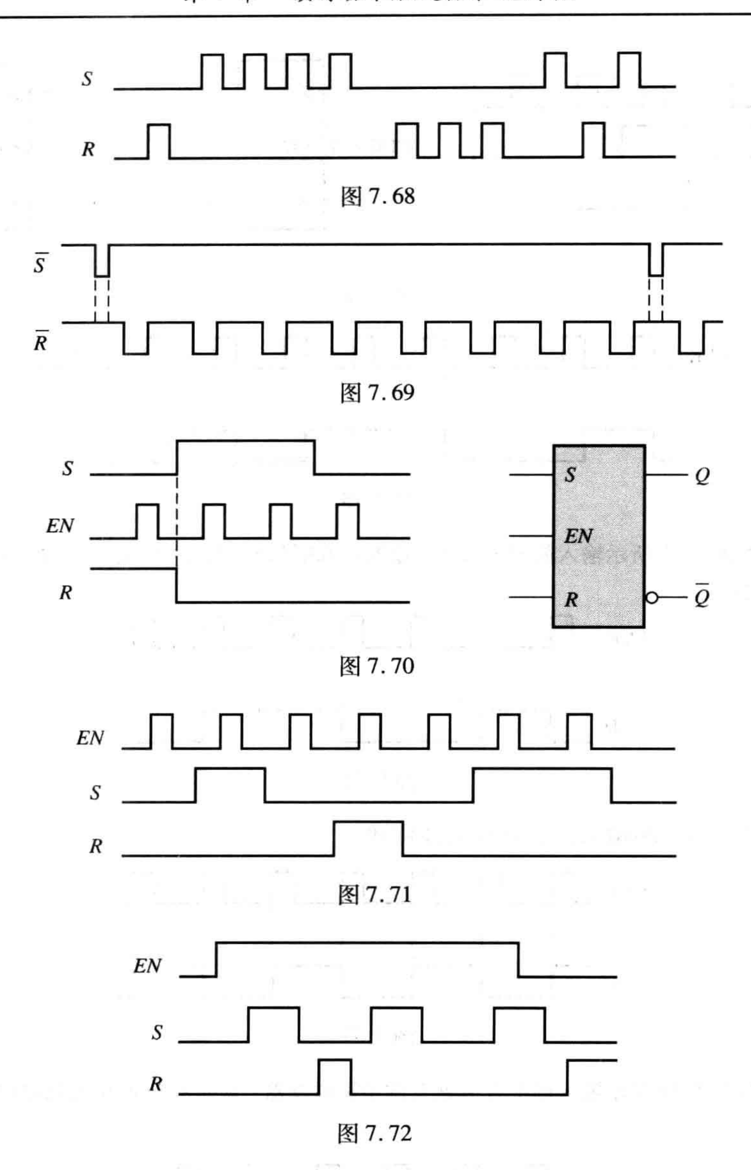


图 7.68

图 7.69

图 7.70

图 7.71

图 7.72

7. 观察到的门控 D 锁存器的输入波形如图 7.73 所示。如果锁存器的初始状态为复位，绘制时序图，指出所期望看到的输出 Q 的波形。

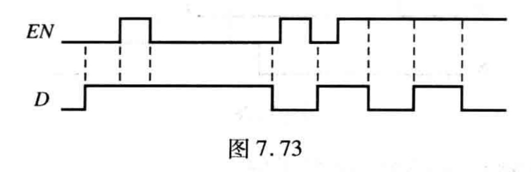


图 7.73

7.2 节　边沿触发器

8. 两个边沿触发的 S-R 触发器如图 7.74 所示。如果输入如图所示的那样，绘制出和时钟关联的每个触发器的 Q 输出，并解释两者之间的区别。触发器初始状态为复位。
9. 边沿触发的 S-R 触发器和时钟信号关联的 Q 输出如图 7.75 所示。确定产生这个输出所需要的 S 和 R 输入上的输入波形，假设触发器为上升沿触发类型。

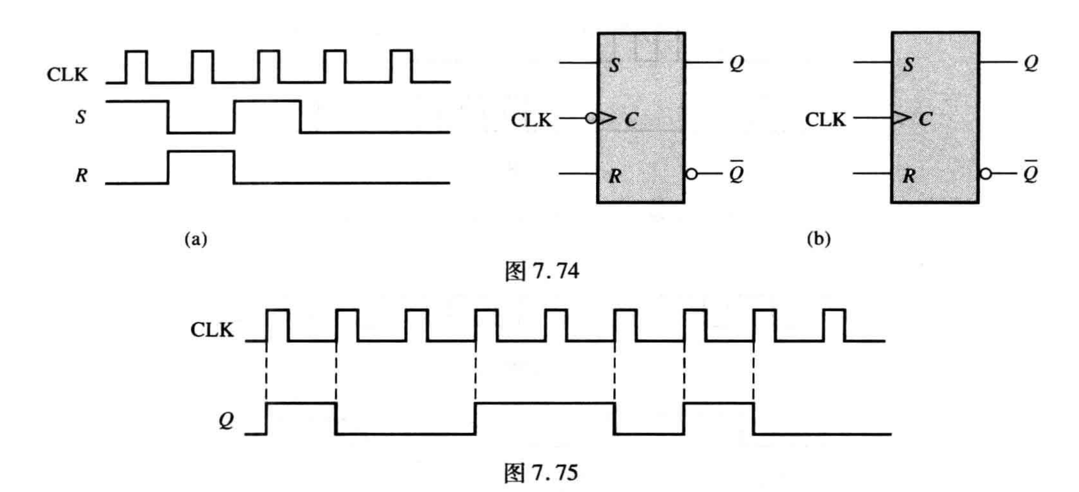


图 7.74

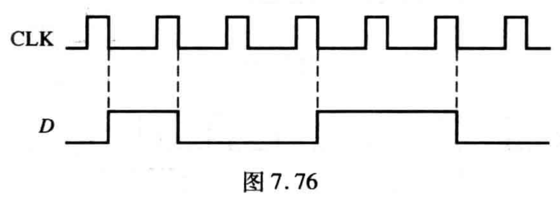


图 7.75

10. 对于具有如图 7.76 所示输入的 D 触发器，绘制出和时钟相关联的 Q 输出。假设为上升沿触发并且 Q 初始为低电平。

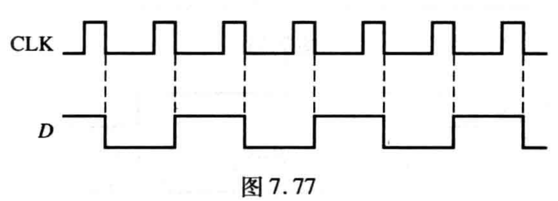


图 7.76

11. 对于如图 7.77 所示的输入，给出如习题 10 的解。

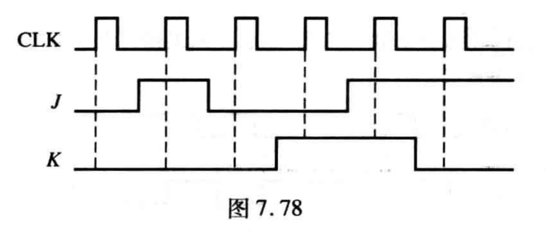


图 7.77

12. 对于具有如图 7.78 所示输入的上升沿触发的 J-K 触发器，确定和时钟相关联的输出。假设 Q 开始于低电平。

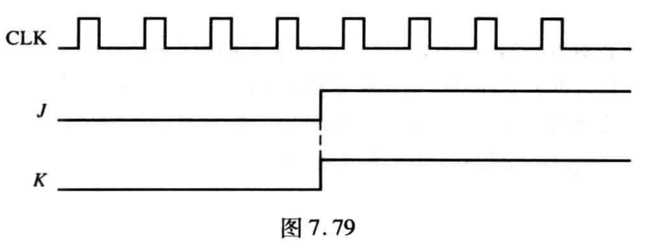


图 7.78

13. 对于如图 7.79 所示的输入，给出如习题 12 的解。

图 7.79

14. 如果如图 7.80 所示的信号加在 J-K 触发器的输入，确定和时钟相关联的 Q 波形。假设 Q 初始为低电平。

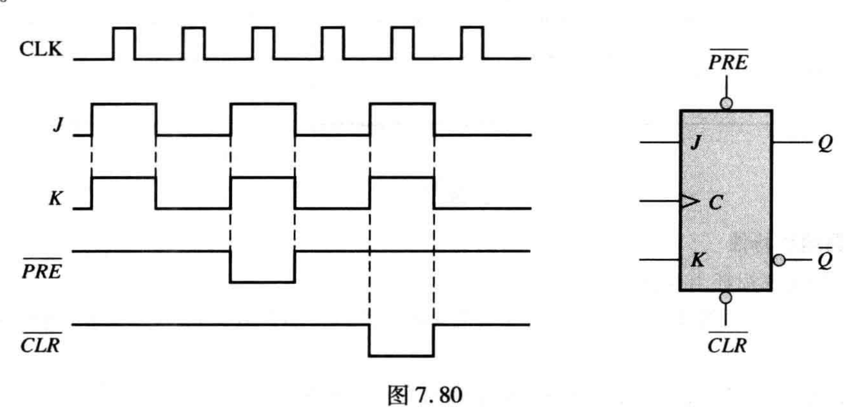


图 7.80

15. 对于具有如图 7.81 所示输入的下降沿触发的 J-K 触发器，给出和时钟相关联的 Q 输出。假设 Q 初始为低电平。

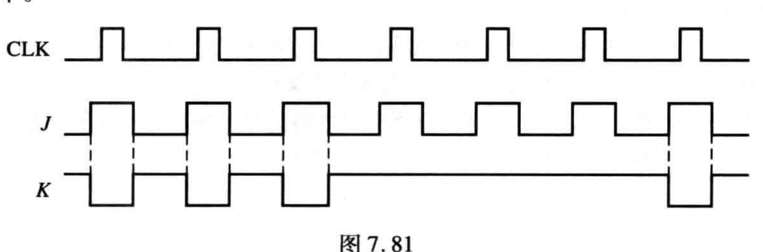


图 7.81

16. 如图 7.82 所示，下面的串行数据通过与门加在触发器上。确定在输出 Q 上所得到的串行数据。每个位时间都有一个时钟脉冲。假设 Q 初始为 0，并且 $\overline{PRE}$ 和 $\overline{CLR}$ 都是高电平。最右边的位首先加入。

J_1：1010011；J_2：0111010；J_3：1111000；K_1：0001110；K_2：1101100；K_3：1010101

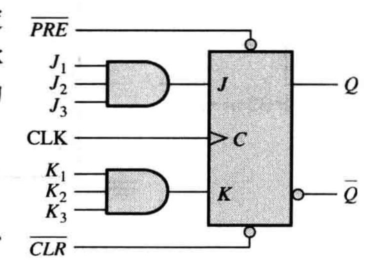


图 7.82

17. 如图 7.82 中的电路，完成图 7.83 的时序图，画出输出 Q(初始为低电平)。假设 $\overline{PRE}$ 和 $\overline{CLR}$ 保持为高电平。

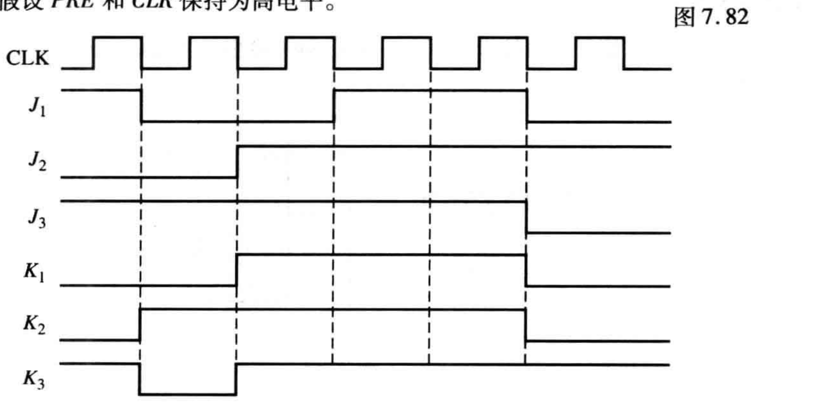


图 7.83

18. 以相同的 J 和 K 输入，但是 $\overline{PRE}$ 和 $\overline{CLR}$ 和时钟的关系如图 7.84 所示，解出如习题 17 的问题。

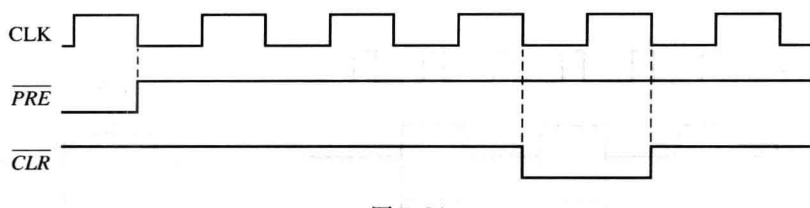


图 7.84

7.3 节 触发器运算特性

19. 触发器的功率损耗由什么决定?
20. 通常，生产商的数据表指定和触发器相关的 4 种不同的传播延迟时间。给出每一种时间的名称并加以描述。
21. 某个触发器的数据表指定时钟脉冲的最小高电平时间为 30 ns，最小低电平时间为 37 ns。那么最大运行频率是多少?
22. 如图 7.85 所示的触发器初始状态为复位。给出输出 Q 和时钟脉冲之间的关系，假设传播延迟 t_{PLH} (时钟脉冲到 Q)是 8 ns。
23. 运行在 +5 V 直流电源上的一个特殊触发器所需要的直流电流为 10 mA。某个数字设备使用 15 个这样的触发器。确定 +5 V 直流电源所需要的电流容量及该系统的总功耗。
24. 对于如图 7.86 所示的电路，确定可靠运行下的最大时钟信号频率，假设每个触发器的建立时间为 2 ns，并且从时钟到输出的延迟时间(t_{PLH}和 t_{PHL})为 5 ns。

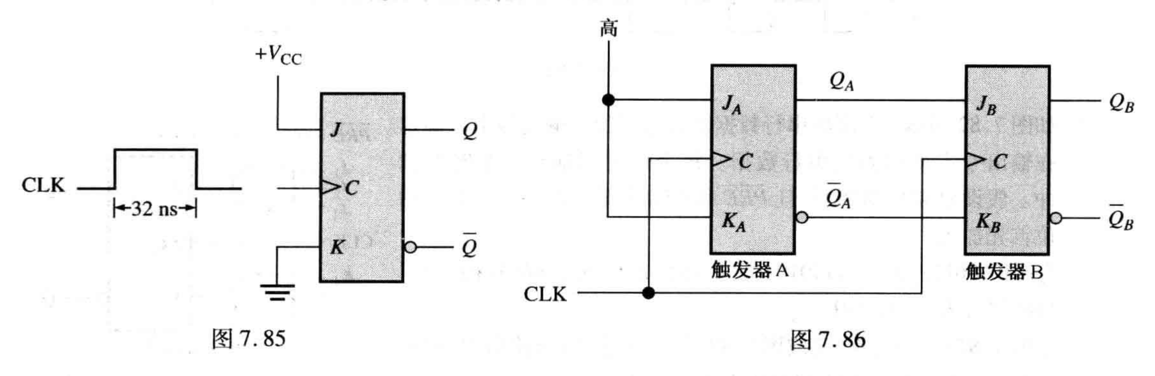


图 7.85　　图 7.86

7.4 节 触发器应用

25. 某个 D 触发器的连接方式如图 7.87 所示。确定和时钟相关联的输出 Q。这个设备有什么样的特殊运行功能?

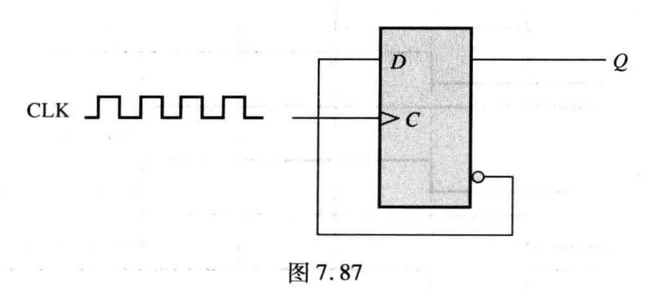


图 7.87

26. 对于图 7.86 中的电路，为 8 个时钟脉冲开发一个时序图，给出和时钟相关联的输出 Q_A 和 Q_B。

7.5 节 单稳态触发器

27. 如果外部电阻为 3.3 kΩ 和外部电容为 2000 pF，确定 74121 单稳态触发器的脉冲宽度。
28. 74LS122 单稳态触发器要产生一个 5 μs 时间间隔的输出脉冲。使用 10 000 pF 的电容，确定所需要的外部电阻的值。
29. 使用 555 定时器组成单稳态触发器，产生一个 0.25 s 的输出脉冲。

7.6 节 非稳态多谐振荡器

30. 把 555 定时器配置成为非稳态多谐振荡器运行，如图 7.88 所示。确定它的频率。

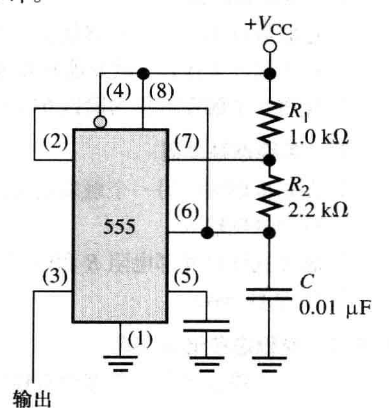


图 7.88

数字系统应用

31. 使用 555 定时器，重新设计交通信号灯控制系统部分的时序电路，产生大约 6 秒的警示黄灯与 40 秒的红灯和绿灯。
32. 使用 74121 单稳芯片，重做习题 31。
33. 使用 74122 单稳芯片，重做习题 31。

特殊设计问题

34. 设计一个基本的计数电路，使用下降沿触发的 J-K 触发器，产生从 0 ~ 7 的二进制序列。
35. 在一个垒球工厂的装箱部门，装运过程为垒球在一个传送带上运行，然后通过一个滑槽按队列进入箱子。每个垒球通过滑槽时启动一个开关电路，开关电路产生一个电脉冲。每个箱子可以放 32 个垒球。设计一个逻辑电路，当箱子装满时，一个空箱子会传送到此位置。
36. 增加一个 15 秒的主干道的左转箭头指示，列出交通信号灯控制系统需要改变的部分。当红灯先于绿灯亮时，箭头指示被点亮。修改第 6 章的状态框图，指出这些变化之处。

Multisim 故障诊断实践

37. 打开文件 P07-45，测试锁存器以确定哪一个损坏。
38. 打开文件 P07-46，测试 J-K 触发器以确定哪一个损坏。
39. 打开文件 P07-47，测试 D 触发器以确定哪一个损坏。
40. 打开文件 P07-48，测试单稳以确定哪一个损坏。
41. 打开文件 P07-49，测试除以 4 电路以确定是否有问题。如果有问题，尽可能查出故障。

答案

温故而知新

7.1 节 锁存器

1. 三种类型的锁存器是 S-R、门控 S-R 和门控 D。
2. $SR=00$，NC；$SR=01$，$Q=0$；$SR=10$，$Q=1$；$SR=11$，无效状态。
3. $Q=1$。

7.2 节 边沿触发器

1. 当使能(EN)端有效时，门控 S-R 锁存器的输出可以发生在任何时间。一个边沿触发的 S-R 触发器的输出只有在时钟触发边沿来到时才能改变。
2. J-K 触发器与 S-R 触发器不一样，它没有无效状态。
3. 第一个脉冲的下降沿输出 Q 变为高电平，第二个脉冲的下降沿输出变为低电平，第三个脉冲的下降沿输出变为高电平，第四个脉冲的下降沿输出变为低电平。

7.3 节 触发器运算特性

1. (a)建立时间为时钟脉冲边沿到来之前，出现输入数据所需要的时间。
   (b)保持时间为时钟脉冲边沿到来之后，保持输入数据所需要的时间。
2. 按照表 7.5，74AHC74 可以工作在最高频率。

7.4 节 触发器应用

1. 存储数据的一组触发器就是寄存器。
2. 对于除以 2 操作，触发器必须处在切换状态($J=1$，$K=1$)。
3. 使用 6 个触发器作为除以 64 电路。

7.5 节 单稳态触发器

1. 在它可以响应另一个触发输入之前，不可重复触发单稳的响应结束(不响应)。可重复触发单稳响应每个触发输入。
2. 脉冲宽度由外部电阻 R 和电容 C 确定。
3. 11 毫秒(ms)。

7.6 节 非稳态多谐振荡器

1. 一个非稳态多谐振荡器没有稳定状态。一个单稳有一个稳定状态。
2. 占空比 = (15 ms/20 ms)100% =75%。

例题的相关问题

7.1 如图 7.5(b)所示，Q 输出相同。

7.2 参见图 7.89。

7.3 参见图 7.90。

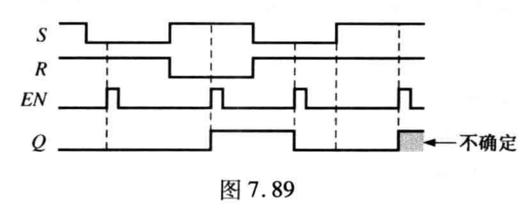


图 7.89

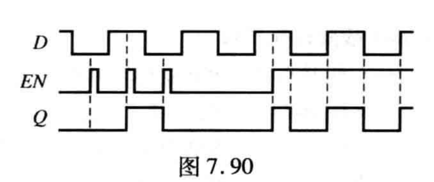


图 7.90

7.4 参见图 7.91。

7.5 参见图 7.92。

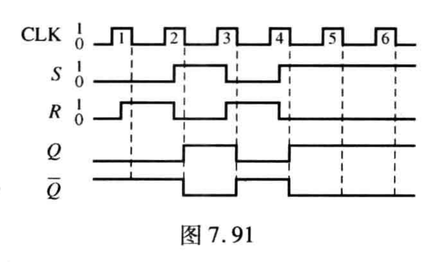


图 7.91

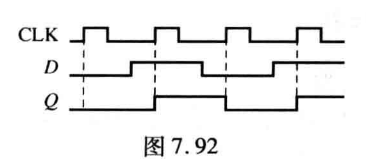


图 7.92

7.6 参见图 7.93。

7.7 参见图 7.94。

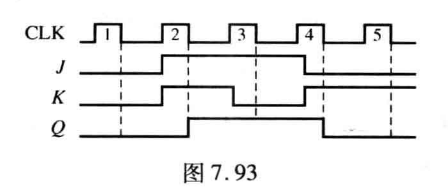


图 7.93

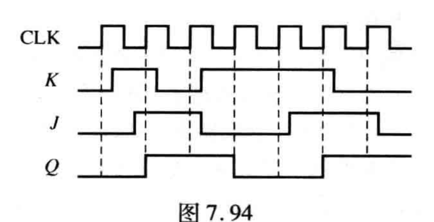


图 7.94

7.8 参见图7.95。

7.9 参见图7.96。

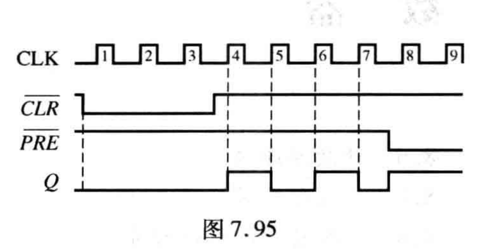


图7.95

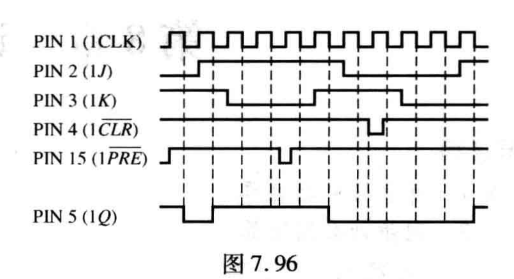


图7.96

7.10 $2^5=32$。需要5个触发器。

7.11 6种状态需要4个触发器($2^4=16$)。

7.12 在74121的 *RX/CX* 到 *CX* 之间连接电容 C_{EXT} =7143 pF，没有接外接电阻。

7.13 C_{EXT} =560 pF，R_{EXT} =27 kΩ，参见图7.97。

7.14 R_1 =91 kΩ。

7.15 占空比≅32%。

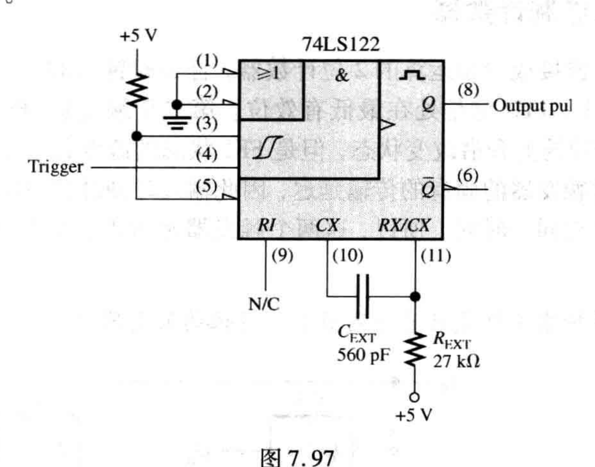


图7.97

判断题

1. T 2. F 3. T 4. T 5. F 6. T 7. T 8. F 9. T 10. T

自测题

1. (a) 2. (c) 3. (d) 4. (b) 5. (d) 6. (d)
7. (a) 8. (b) 9. (d) 10. (d) 11. (c) 12. (f)

第8章　计　数　器

章节提纲

8.1　异步计数器运算

8.1.1　2位异步二进制计数器

图8.1给出了一个连接成异步运算的2位计数器。注意时钟(CLK)只应用于第一个触发器(FF0)的时钟输入(C)上，FF0总是处在最低有效位。第二个触发器(FF1)由FF0的输出来触发。FF0在每个时钟脉冲的上升沿改变状态，但是FF1状态的改变仅发生在FF0输出的上升沿转换的时间。由于通过触发器的固有的传输延迟，因此输入时钟脉冲(CLK)的转换和FF0输出的转换绝对不可能发生在同一时间。所以，这两个触发器永远不会同时被触发，因而该计数器的运算是异步的。

◇ 异步计数器的时钟输入总是只连接到最低有效位的触发器上。

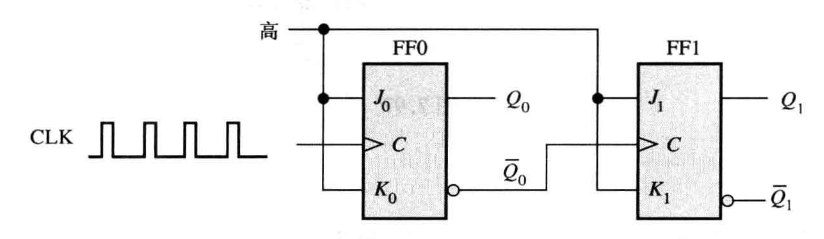


图8.1　2位异步二进制计数器。打开文件F08-01检验运行结果

时序图　通过加入4个时钟脉冲到FF0，观察每个触发器的Q输出，检查一下图8.1中异步计数器的基本运算。图8.2解释了对应于时钟脉冲的触发器输出状态的变化。两个触发器连接为切换运算($J=1$，$K=1$)，并且假设它们初始状态为复位(Q为低电平)。

◇ 异步计数器也称为行波(前级的输出连接后级的输入，信号的传送就像波的行进)计数器。

CLK1(时钟脉冲1)的上升沿使得FF0的Q_0输出变为高电平，如图8.2所示。与此同时输出$\overline{Q}_0$变为低电平，但是它对FF1没有影响，因为必须发生上升沿的转换才使触发器触发。在CLK1的上升沿之后，$Q_0=1$和$Q_1=0$。CLK2的上升沿使得Q_0变为低电平。输出$\overline{Q}_0$变为低电平，而触发器FF1使得Q_1变为高电平。在CLK2的上升沿之后，$Q_0=0$和$Q_1=1$。CLK3的上升

沿再次使得 Q_0变为高电平。输出 $\bar{Q}_0$ 变为低电平，并且对 FF1 没有影响。因此，在 CLK3 的上升沿之后，$Q_0=1$ 和 $Q_1=1$。CLK4 的上升沿使得 Q_0变为低电平，同时 $\bar{Q}_0$ 变为高电平，而触发器 FF1 使得 Q_1 变为低电平。在 CLK4 的上升沿之后，$Q_0=0$ 和 $Q_1=0$。现在计数器已经再循环到了它的初始状态(两个触发器都是复位状态)。

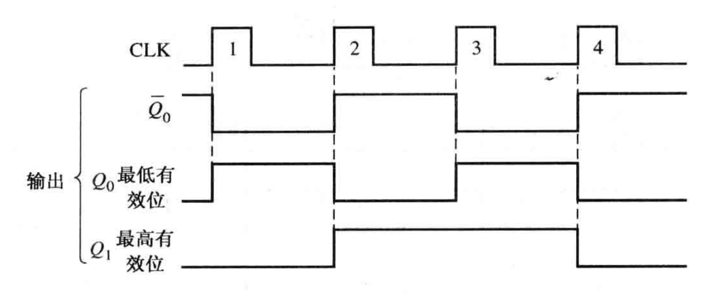


图 8.2 图 8.1 中计数器的时序图

在时序图中，Q_0和 Q_1输出的波形如图 8.2 所示，它们和时钟脉冲关联。为了简化的目的，Q_0、Q_1和时钟脉冲的转换以同步形式给出，尽管这是个异步计数器。当然，在 CLK 和 Q_0转换之间及在 $\bar{Q}_0$ 和 Q_1的转换之间，有一些小的延迟。

注意在图 8.2 中，2 位计数器呈现了 4 种不同的状态，正如对于两个触发器($2^2=4$)所期望的那样。当然，注意如果 Q_0表示最低有效位而 Q_1表示最高有效位，计数器状态的顺序表示如表 8.1 所列出的一系列二进制数。

表 8.1 图 8.1 中计数器的二进制状态序列

时钟输入	Q_1	Q_0
初始状态	0	0
1	0	1
2	1	0
3	1	1
4(再循环)	0	0

◇ *在数字逻辑中，Q_0总是最低有效位，除非特别说明。*

由于计数器经历了一个二进制序列，所以图 8.1 中的计数器是一个二进制计数器。它实际上计数了 3 个时钟脉冲，而在第 4 个脉冲上，再循环到它的初始状态($Q_0=0$，$Q_1=0$)。术语再循环(recycle)一般用在计数器运算上，它是指计数器从它的最终状态返回到初始状态的转换。

8.1.2 3 位异步二进制计数器

表 8.2 列出了 3 位二进制计数器的状态时序，3 位二进制异步计数器如图 8.3(a)所示。该基本运算和 2 位计数器的运算是一样的，只不过由于有 3 个触发器，3 位计数器有 8 个状态。8 个时钟脉冲的时序图如图 8.3(b)所示。注意，计数器经过一个从 0 ~ 7 的二进制计数，然后再循环回到 0 状态。通过连接附加的切换状态的触发器，可以很容易地将这个计数器扩展为更多位的计数。

表 8.2 3 位二进制计数器的状态序列

时钟输入	Q_2	Q_1	Q_0
初始状态	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8(再循环)	0	0	0

传输延迟 由于下述原因，异步计数器常常称为行波(ripple)计数器：输入时钟脉冲的效应首先被 FF0“感觉”到。由于通过 FF0 的传输延迟，这个效应不能立即到达 FF1。然后，在 FF2 被触发之前，还有一个通过 FF1 的传输延迟。因此，输入时钟脉冲的效应以“行波”的形式通过计数器，由于传输延迟，花费了一些时间之后才能使输入效应到达最后一个触发器。

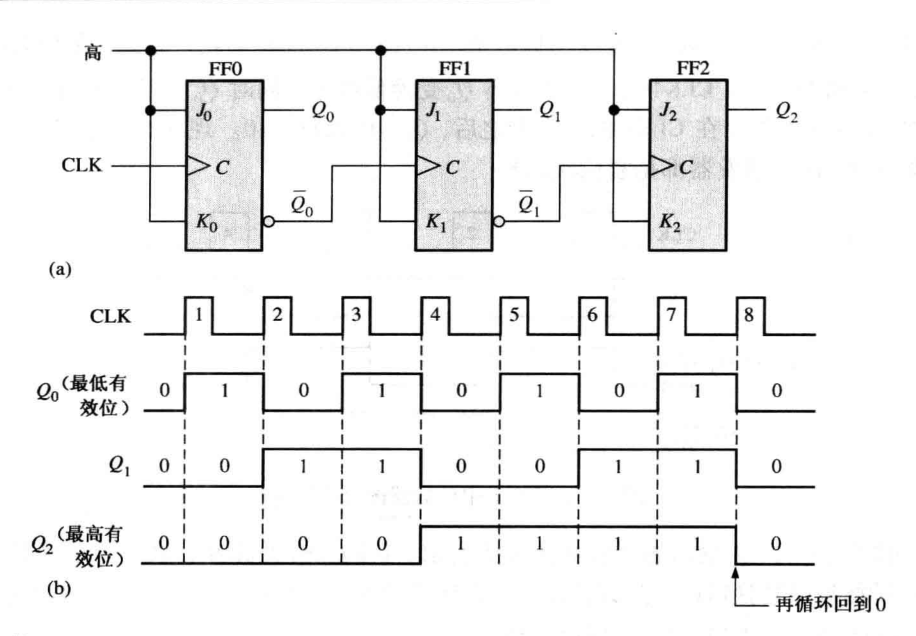


图 8.3 3 位异步二进制计数器和一个周期的时序图。打开文件 F08-03 检验运行结果

为了说明这一点，注意图 8.3 中计数器的所有三个触发器都在 CLK4 的上升沿改变状态。这种行波时钟效应如图 8.4 所示，其中给出了前 4 个时钟脉冲，同时指出了传输延迟。Q_0从高电平到低电平的转换在时钟脉冲的上升沿转换后一个延迟时间(t_{PLH})内发生。Q_1从高电平到低电平的转换在时钟脉冲的上升沿转换之后一个延迟时间(t_{PLH})内发生。Q_2从高电平到低电平的转换在时钟脉冲的上升沿转换之后一个延迟时间(t_{PLH})内发生。正如所看到的那样，FF2 直到时钟脉 CLK4 上升沿之后的两个延迟才被触发。因此，时钟脉冲 CLK4 的效应要花费三个传输延迟时间以“行波”的形式通过计数器，然后才能使 Q_2从低电平变为高电平。

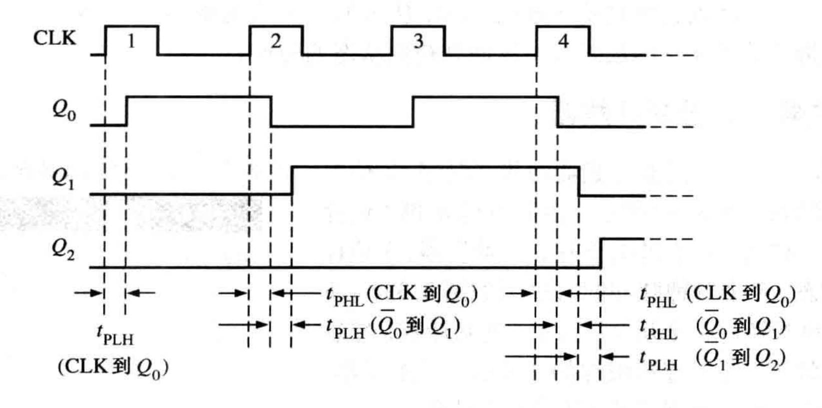


图 8.4 3 位异步(行波时钟)二进制计数器的传输延迟

异步计数器的这种积累延迟在许多应用中都是一个主要的缺点，因为它限制了计数器按时钟运行的速度，并且产生了译码问题。计数器中的最大积累延迟必须小于时钟波形的周期。

例 8.1 4 位异步二进制计数器如图 8.5(a)所示。每个触发器都是下降沿触发，并且具有传输延迟 10 ns。画出一个时序图，给出每个触发器的输出 Q，并且确定从时钟脉冲的触发边沿到 Q_3状态发生相应变化之间的总传输延迟时间。同时确定该计数器可以运行的最大时钟频率。

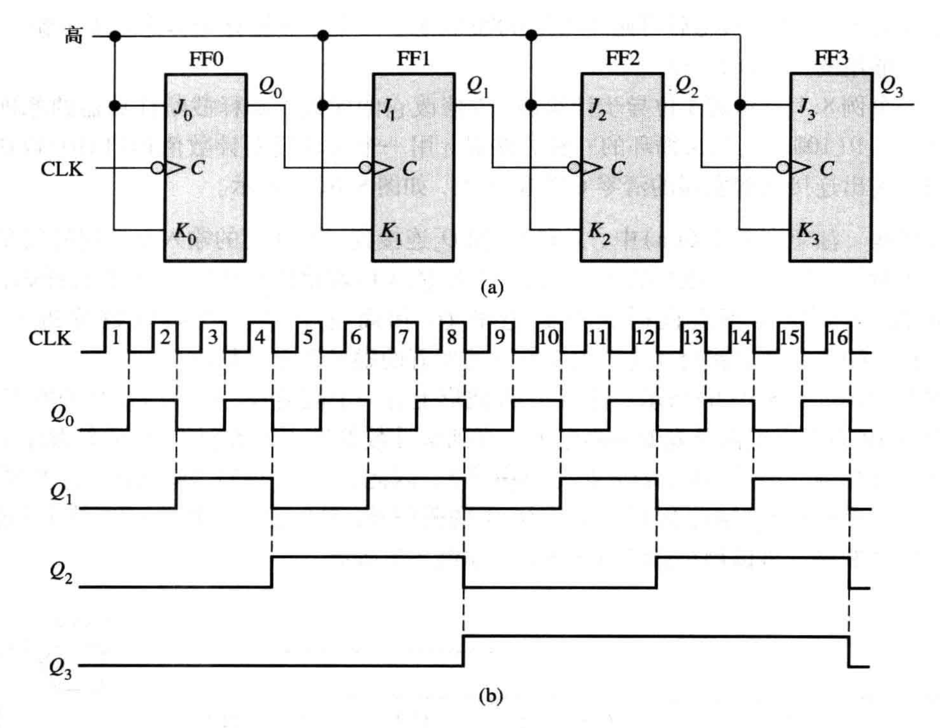


图 8.5 4 位异步二进制计数器和时序图。打开文件 F08-05 检验运行结果

解：忽略延迟的时序图如图 8.5(b)所示。对于总延迟时间，CLK8 或者 CLK16 的效应必须在 Q_3 改变之前通过 4 个触发器，所以

$$t_{p(\text{总时间})} = 4 \times 10\ \text{ns} = \mathbf{40\ ns}$$

最大时钟频率为

$$f_{\max} = \frac{1}{t_{p(\text{总时间})}} = \frac{1}{40\ \text{ns}} = \mathbf{25\ MHz}$$

相关问题①：如果图 8.5(a)中的所有触发器都是上升沿触发的，请画出时序图。

8.1.3 异步译码计数器

◇ 计数器可以有 2^n 个状态，其中 n 是触发器的个数。

计数器的模是计数器按顺序经过的独特状态的数目。计数器的最大可能状态(最大模)是 2^n，其中 n 是计数器内部的触发器个数。当然，可以设计计数器，使得其序列中的状态个数小于最大值 2^n。这种序列类型称为截断序列。

一个具有截断序列的计数器的常见模是 10(称为模 10)。序列中具有 10 个状态的计数器称为十进制计数器。具有从 0(0000) ~ 9(1001)计数序列的十进制计数器是 BCD 十进制计数器，因为它的 10 种状态序列产生 BCD 码。这种类型的计数器在一些显示应用中很有用，使用中需要将 BCD 码变换为十进制读数。

为了得到截断的序列，需要迫使计数器在经过所有可能的状态之前再循环。例如，BCD 十

---

① “相关问题”的答案请参见本章结尾。

进制计数器必须在 1001 状态后再循环回到 0000 状态。一个十进制计数器需要 4 个触发器(3 个触发器是不够用的，因为 $2^3=8$)。

使用一个例 8.1 所示的 4 位异步计数器，并修改它的序列以解释截断计数器的原理。使计数器在计数到 9(1001)之后再循环的一种方法是，用一个与非门对计数值10(1010)译码，并把此与非门的输出连接到触发器的清零 $\overline{CLR}$ 输入上，如图 8.6(a)所示。

部分译码　注意在图 8.6(a)中，只有 Q_1 和 Q_3连接到了与非门的输入上。这种安排是部分译码的一个例子，其中两个独特的状态($Q_1=1$ 和 $Q_3=1$)就足以对计数值 10 进行译码，因为其他所有状态(0 ~ 9)都不会同时具有高电平 Q_1 和高电平 Q_3。当该计数器进入计数值 10(1010)时，译码门输出就会变为低电平并且使所有的触发器异步复位。

结果时序图如图 8.6(b)所示。注意在 Q_1 波形上有一个假信号。产生假信号的原因是 Q_1 必须在计数值 10 被译码之前首先变为高电平。直到该计数器进入计数值 10 以后的几纳秒，译码门的输出才会变为低电平(两个输入都是高电平)。因此，在复位到 0000 之前，计数器在 1010 状态上停留一个较短的时间，因而产生了 Q_1 上的假信号，导致复位计数器 $\overline{CLR}$ 线上的假信号。

其他的截断序列可以用相似的方式实现，如例 8.2 所示。

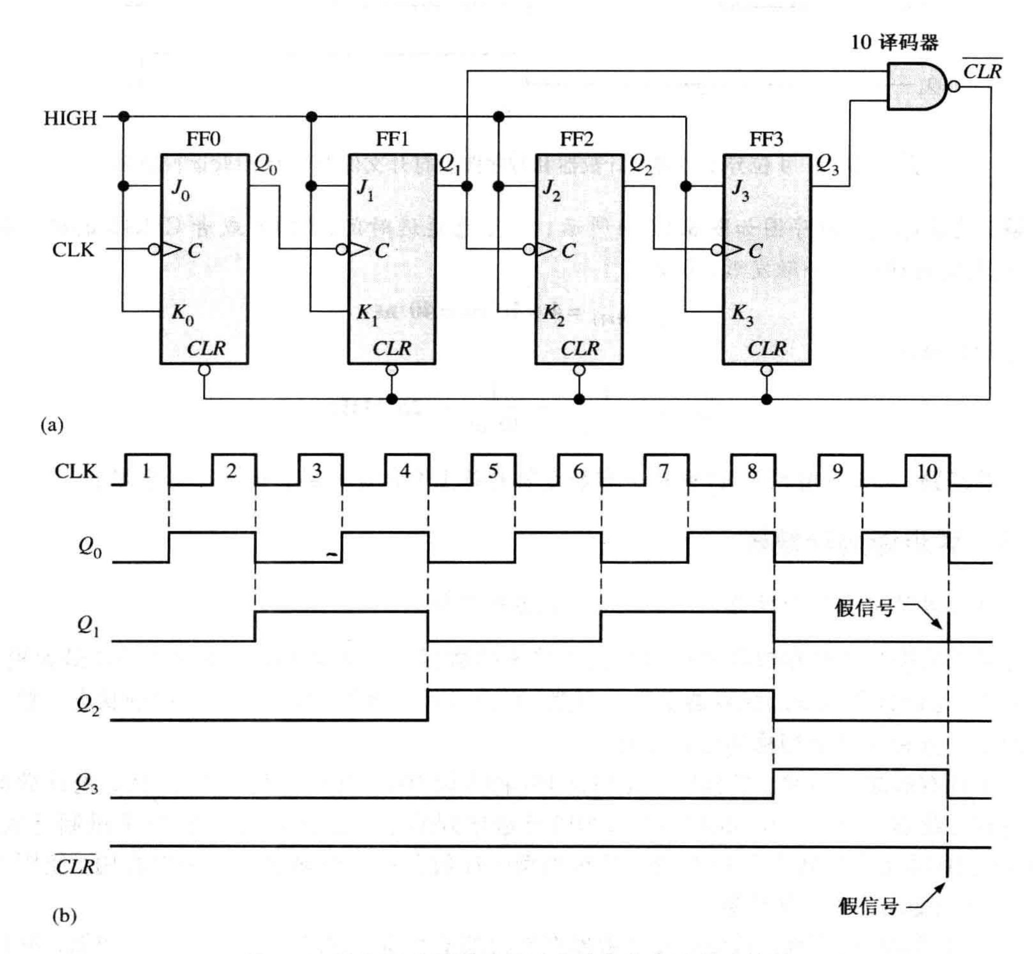
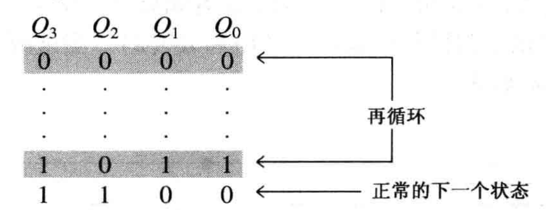


图 8.6　具有异步再循环的异步触发的十进制计数器

例8.2 给出怎样实现一个异步计数器，使其模为12，具有从0000到1011的直接二进制序列。

解： 由于3个触发器可以产生最多8个状态，所以就需要4个触发器来产生任何大于8而小于或者等于16的模。

当该计数器进入它的最后一个状态1011后，它就必须再循环到0000而不是进入其正常的下一个状态1100，如下面的序列图所示：



观察Q_0和Q_1都变为0，但是Q_2和Q_3都必须在第12个时钟脉冲时被迫转换为0。如图8.7(a)给出了这个模12计数器。与非门部分译码计数值12(1100)并且使触发器2和触发器3复位。因此，在第12个时钟脉冲时，计数器被迫从计数值11再循环回到计数0，如图8.7(b)中的时序图所示。(在被$\overline{CLR}$上的假信号复位之前，计数器只在计数值12上停留几纳秒的时间。)

相关问题： 怎样修改图8.7(a)中的计数器使其变为模13计数器？

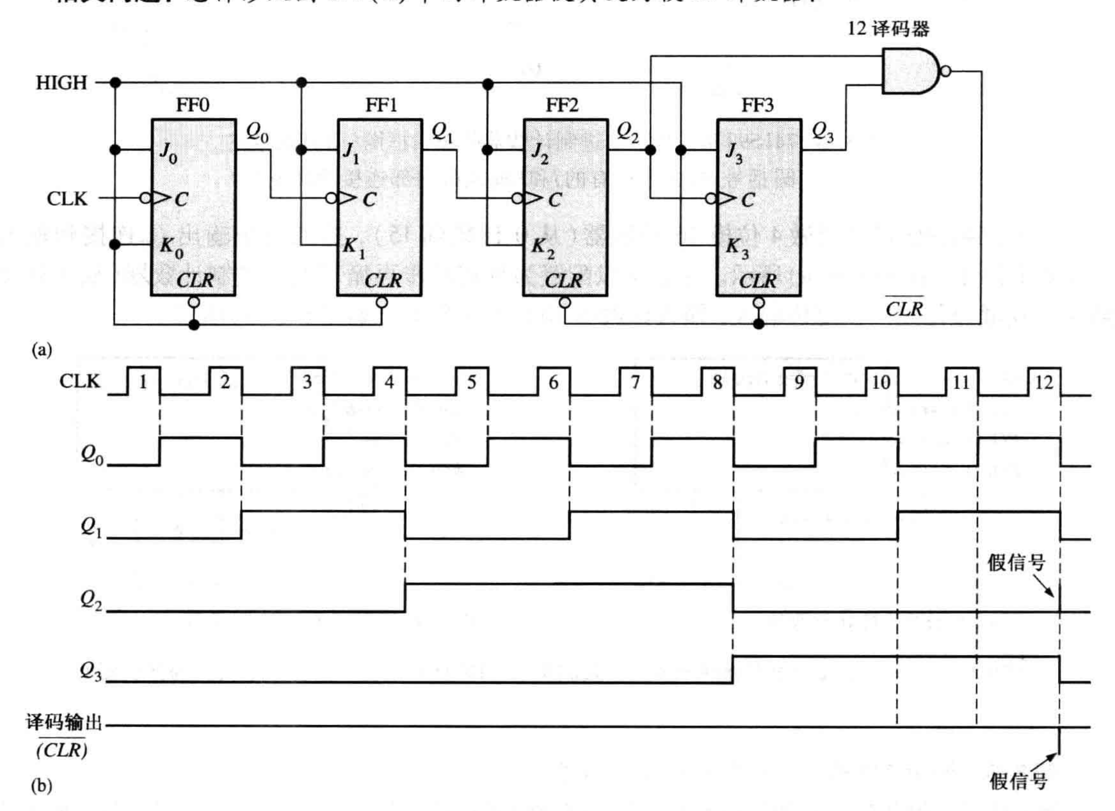


图8.7 具有异步再循环的异步触发的模12计数器



74LS93 4 位异步二进制计数器

74LS93 是特定集成电路异步计数器的一个例子。如图 8.8 中逻辑框图所给出的那样，这种芯片实际上由一个单一的触发器和一个 3 位异步计数器组成。这种安排是出于灵活的目的。如果只使用单个触发器，它可以用做除以 2 的电路；或者如果只使用 3 位计数器，它可以用做模 8 计数器。这块芯片同时提供了门控复位输入 $RO(1)$ 和 $RO(2)$。当这两个输入都是高电平时，计数器就将 $\overline{CLR}$ 复位到 0000 状态。

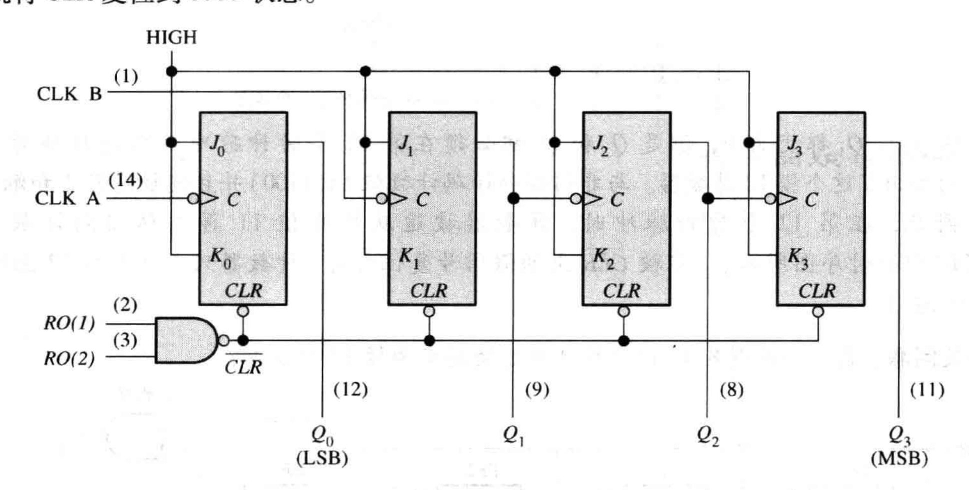


图 8.8 74LS93 4 位异步二进制计数器的逻辑框图(引脚编号在圆括号内,并且所有的J和K输入都内部连接为高电平)

此外，74LS93 可以用做 4 位模 16 计数器(从 0 计数到 15)，这通过把输出 Q_0 连接到输入 CLK B 上即可，如图 8.9(a)所示。它也可以配置为具有异步再循环的十进制计数器(从 0 计数到 9)，这通过使用门控复位输入，输入计数值 10 的部分译码，如图 8.9(b)所示。

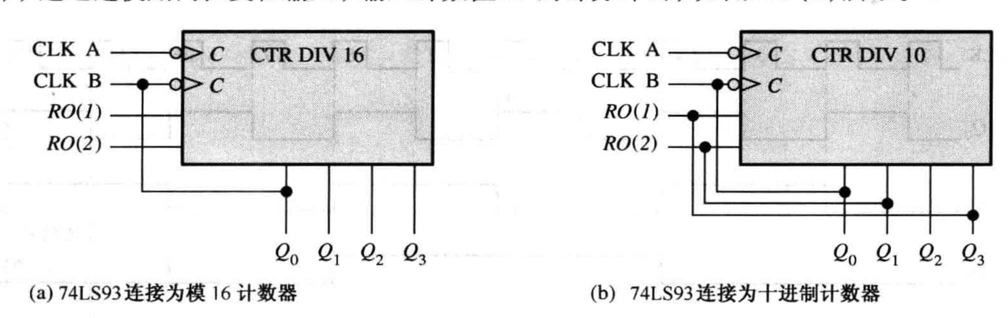


图 8.9 74LS93 异步计数器的两种配置(限制符号 CTR DIV n 表示具有 n 个状态的计数器)

例 8.3 给出 74LS93 怎样用做模 12 计数器。

解: 使用门控复位输入 $RO(1)$ 和 $RO(2)$，部分译码计数值 12(记住，有一个内部与非门和这两个输入相关联)。计数值 12 的译码通过把 Q_3 连接到 $RO(1)$ 和 Q_2 连接到 $RO(2)$ 来完成，如图 8.10 所示。输出 Q_0 连接到 CLK B 上，最后形成一个 4 位计数器。

在计数器进入计数值12(1100)之后的瞬间，它就会复位到0000。但是，这个再循环会在Q_2上产生一个假信号，因为该计数器在再循环之前，必须进入1100状态并停留几纳秒的时间。

相关问题： 给出74LS93怎样连接为模13计数器。

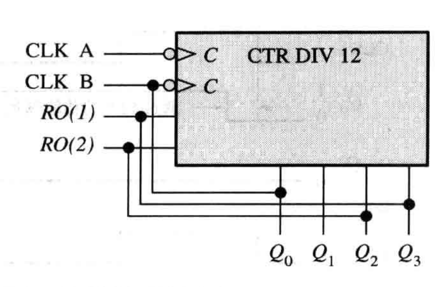


图8.10 74LS93连接为模12计数器

8.1节 温故而知新 (答案请参见本章结尾。)

1. 异步计数器中的异步指的是什么？

2. 一个模14计数器有多少状态？需要至少多少个触发器来实现它？

8.2 同步计数器运算

8.2.1 2位同步二进制计数器

图8.11给出了一个2位同步二进制计数器。注意必须为FF1的J_1和K_1输入使用不同于异步计数器的排列，以实现二进制时序。

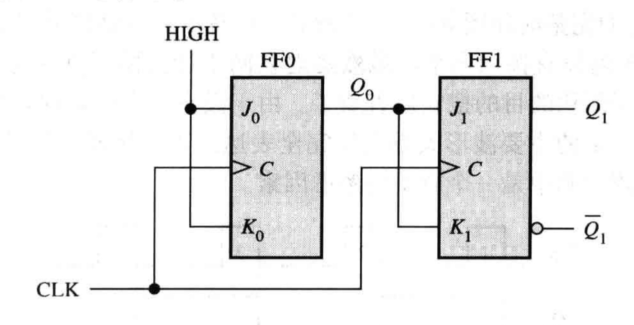


图8.11 2位同步二进制计数器

同步计数器的运算如下所示：首先，假设此计数器的初始状态是二进制0，也就是两个触发器都是复位状态。当第一个时钟脉冲的上升沿到来时，FF0将切换，因而Q_0变为高电平。那么FF1在CLK1的上升沿到来时会发生什么情况呢？为了找出答案，看看FF1的输入条件。因为输入J_1和K_1都连接到Q_0，而Q_0还没有达到高电平，所以J_1和K_1都是低电平。记住，从时钟脉冲的触发边沿到Q输出产生实际上的转换，这之间存在传输延迟。所以，当第一个时钟脉冲的上升沿到来时，$J=0$和$K=0$。这是个无变化情况，所以FF1没有改变状态。计数器运算部分的时序细节如图8.12(a)所示。

在同步计数器中，时钟脉冲作用在每一个触发器上。

在CLK1之后，$Q_0=1$和$Q_1=0$(这是二进制1状态)。当CLK2的上升沿到来时，FF0将切换，并且Q_0将变为低电平。由于FF1在这个时钟脉冲的触发边沿上时，输入J_1和K_1都是高电平($Q_0=1$)，因此触发器切换，并且Q_1变为高电平。由于FF1为高电平($Q_0 = 1$)，并且在这个时钟脉冲的触发边沿时$J_1 = 1$和$K_1 = 1$，所以在CLK2之后，$Q_0=0$和$Q_1=1$(这是二进制2状态)。这个情况的时序细节如图8.12(b)所示。

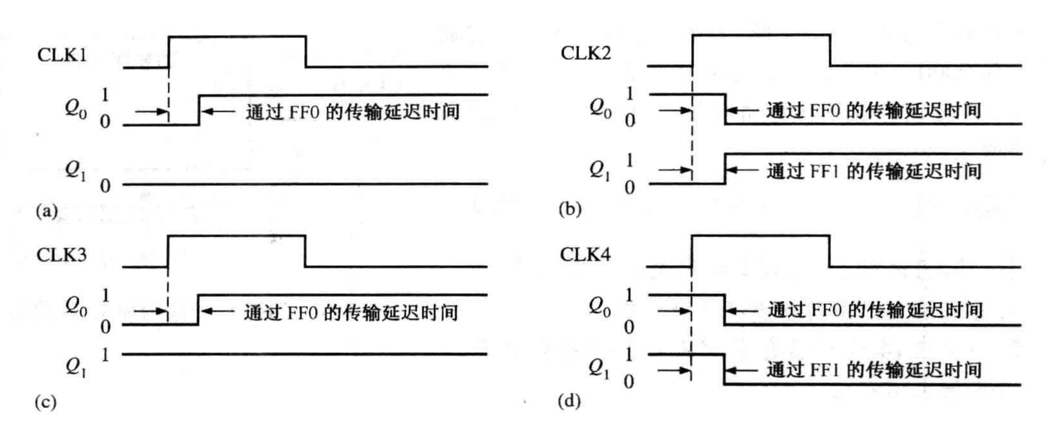


图 8.12　2 位同步计数器运算部分的时序细节(两个触发器的传输延迟时间假设相等)

当 CLK3 的上升沿到来时，FF0 再次切换到置位状态($Q_0=1$)，而 FF1 保持置位($Q_1=1$)，因为它的 J_1 和 K_1 输入都是低电平($Q_0=0$)。在这个触发边沿之后，$Q_0=1$ 和 $Q_1=1$(这是二进制 3 状态)。时序细节如图 8.12(c)所示。

最后，在 CLK4 的上升沿，Q_0 和 Q_1 变为低电平，因为它们在 J 和 K 输入上都处在切换状态。时序细节如图 8.12(d)所示。这时计数器再循环回到它的初始状态，即二进制 0。

图 8.11 中计数器的完整时序图如图 8.13 所示。注意所有的波形转换看起来都是同时发生的；也就是说，传输延迟没有指示出来。虽然延迟在同步计数器运算中是一个重要的因素，但是在整个时序图中出于简化的目的把它们省略了。由于没有给出很短的延迟和非常小的时序差别，电路正常运算所产生的主要波形关系可以完全表达出来。然而，在高速数字电路中，这些小的延迟在设计和故障检测中是一个重要的考虑因素。

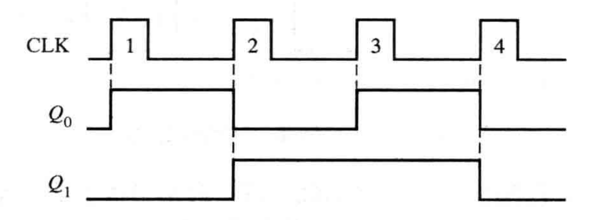


图 8.13　图 8.11 计数器的时序图

8.2.2　3 位同步二进制计数器

一个 3 位同步二进制计数器如图 8.14 所示，它的时序图如图 8.15 所示。通过观察如表 8.3 所示的状态时序，就可以理解这个计数器的运算。

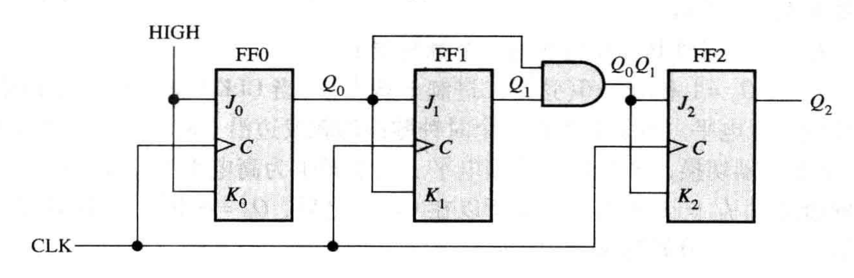


图 8.14　3 位同步二进制计数器。打开文件 F08-14 观察运行结果

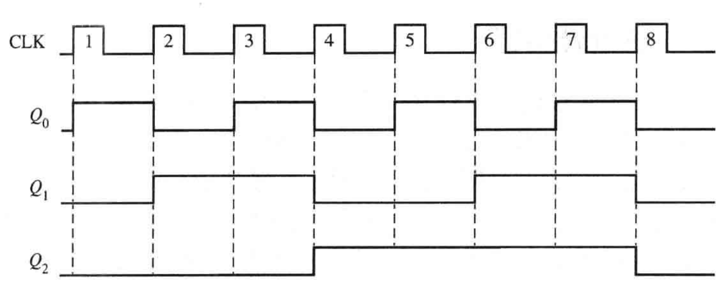


图 8.15 图 8.14 中计数器的时序图



计算机小知识

奔腾系统中的 TSC(时间戳计数器)用来进行性能监测，它可以正确地确定一些对于奔腾系统的整体性能很重要的参数。它通过阅读一个程序执行前后的 TSC，基于处理器循环时间就可以确定此程序所需要的精确时间。在这种方式中，TSC 构成了所有时间计算的基础，这些时间计算和系统运算的优化相关联。例如，可以准确确定两个或者多个程序序列中哪一个更加有效。这对于编译程序开发人员和系统程序员为奔腾系统开发最有效的代码非常有用。

表 8.3 3 位二进制计数器的二进制状态时序

时钟脉冲	Q_2	Q_1	Q_0
初始状态	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8(再循环)	0	0	0

首先，观察 Q_0。注意到随着计数器从它的初始状态变化到最后一个状态然后再返回到初始状态，Q_0在每个时钟到来时都改变状态。与此同时计数器从它的初始状态进入到最终状态，然后再返回它的初始状态。为了产生这个运算，FF0 必须保持在切换模式下，这通过使输入 J_0和K_0上连接固定不变的高电平来实现。注意 Q_1在每次 Q_0为 1 之后，都进入相反的状态。这种改变发生在 CLK2、CLK4、CLK6 和 CLK8 上。CLK8 脉冲使得计数器再循环，为了产生这个运算，Q_0连接到 FF1 的 J_1和 K_1输入上。当 Q_0为 1 和时钟脉冲到来时，FF1 就处于切换模式，因此改变状态。其他时间，当 Q_0为 0 时，FF1 就处于无变化模式，并且保持它当前的状态。

接下来，看看 FF2 怎样依据二进制时序而在恰当的时间改变状态。注意 Q_2的两次状态改变，在这之前都会有一个特定的条件，即 Q_0和 Q_1都是高电平。这个条件由与门来检测，并且应用于 FF2 的 J_2和 K_2输入上。只要 Q_0和 Q_1都为高电平，与门的输出就会使得 FF2 的 J_2和 K_2输入为高电平，并且 FF2 在下一个时钟脉冲到来时切换。在所有的其他时间，J_2和 K_2输入都被与门输出保持为低电平，这样 FF2 的状态不会改变。

图 8.14 的计数器的分析由表 8.4 归纳总结。

表 8.4 图 8.14 的归纳总结

时钟脉冲	输出			*J-K* 输入						下一个时钟脉冲		
	Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0	FF2	FF1	FF0
初始状态	0	0	0	0	0	0	0	1	1	无变化	无变化	切换
1	0	0	1	0	0	1	1	1	1	无变化	切换	切换
2	0	1	0	0	0	0	0	1	1	无变化	无变化	切换
3	0	1	1	1	1	1	1	1	1	切换	切换	切换
4	1	0	0	0	0	0	0	1	1	无变化	无变化	切换
5	1	0	1	0	0	1	1	1	1	无变化	切换	切换
6	1	1	0	0	0	0	0	1	1	无变化	无变化	切换
7	1	1	1	1	1	1	1	1	1	切换	切换	切换
										计数器再循环回到 0000		

8.2.3 4 位同步二进制计数器

图 8.16(a)给出了一个 4 位同步二进制计数器，图 8.16(b)给出了它的时序图。这个特殊的计数器由下降沿触发的触发器来实现。对于前 3 个触发器，J 和 K 输入控制的内在推导和前面所讨论的 3 位计数器是一样的。第四个阶段，FF3 在序列中只改变了两次。注意这两个转换的发生都在 Q_0、Q_1 和 Q_2 全部为高电平之后。这个情况由与门 G_2 来译码，使得当时钟脉冲到来时，FF3 就会改变状态。对于所有其他时间，FF3 的 J_3 和 K_3 输入为低电平，并且处于无变化情况。

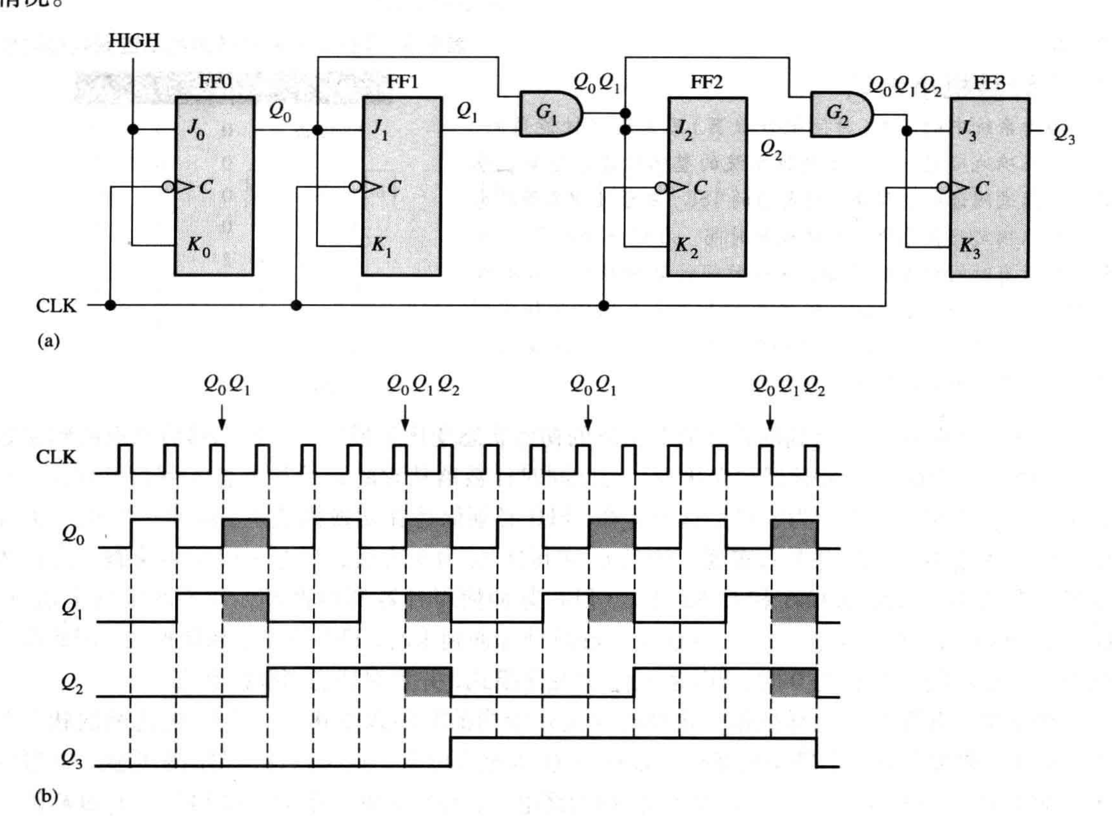


图 8.16 一个 4 位同步二进制计数器和时序图。与门输出为高电平的地方由阴影区域来表示

8.2.4 4 位同步十进制计数器

正如所知道的那样，BCD 十进制计数器呈现出一个截断的二进制序列，从 0000 到 1001 状态，再从 1001 状态循环回到 0000 状态，而不是从 1001 状态到 1010 状态。同步 BCD 十进制计数器如图 8.17 所示。十进制计数器的时序图如图 8.18 所示。

◇ 十进制计数器具有 10 个状态。

通过检查表 8.5 中的状态序列并且遵循图 8.17 中的实现方法，就可以理解该计数器的运算。首先，注意 FF0(Q_0)在每一个时钟脉冲到来时切换，所以 J_0 和 K_0 输入的逻辑等式为

$$J_0 = K_0 = 1$$

这个等式通过把 J_0 和 K_0 连接到固定不变的高电平上来实现。

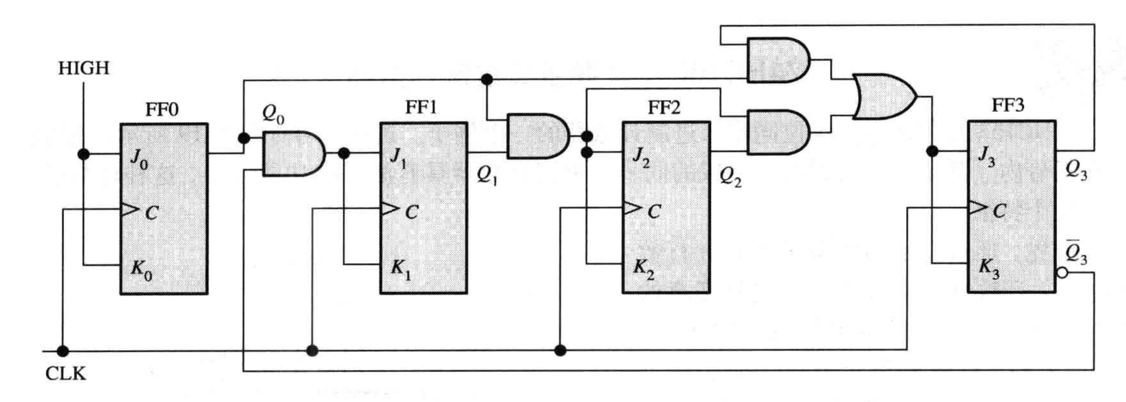


图 8.17 同步 BCD 十进制计数器。打开文件 F08-17 检验运行结果

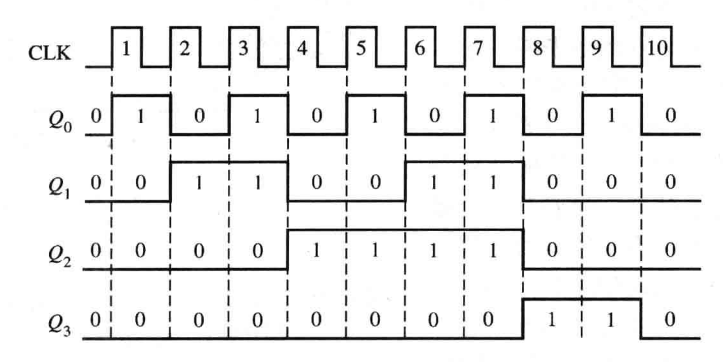


图 8.18 BCD 十进制计数器的时序图(Q_0是最低有效位)

接下来，注意在表 8.5 中，FF1(Q_1)每次在 $Q_0=1$ 和 $Q_3=0$ 的下一个时钟脉冲上改变状态，所以 J_1和 K_1输入的逻辑等式为

$$J_1=K_1=Q_0\bar{Q}_3$$

这个等式由 Q_0和 $\bar{Q}_3$ 相与并且把与门输出连接到 FF1 的 J_1和 K_1输入上来实现。

触发器 2(Q_2)每次在 $Q_0=1$ 和 $Q_1=1$ 的下一个时钟脉冲改变状态。这需要如下所示的输入逻辑等式：

$$J_2=K_2=Q_0Q_1$$

这个等式由 Q_0和 Q_1相与并且把门输出连接到 FF2 的 J_2和 K_2输入上来实现。

最后，FF3(Q_3)每次在 $Q_0=1$、$Q_1=1$ 和 $Q_2=1$(状态 7)的下一个时钟脉冲变为相反的状态，或者在 $Q_0=1$ 和 $Q_3=1$(状态 9)时改变，所对应的等式如下所示：

$$J_3=K_3=Q_0Q_1Q_2+Q_0Q_3$$

这个函数由连接到 FF3 的 J_3和 K_3输入上的与/或逻辑来实现，如图 8.17 中的逻辑框图所示。注意这个十进制计数器和图 8.16 中的模 16 二进制计数器的区别是 Q_0Q_3与门、$Q_0\bar{Q}_3$与门和或门；这种安排检测 1001 状态的发生，并且使得此计数器在下一个时钟脉冲到来时正确地再循环。

表 8.5 BCD 十进制计数器的状态

时钟脉冲	Q_3	Q_2	Q_1	Q_0
初始状态	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10(再循环)	0	0	0	0



74HC163 4位同步二进制计数器

74HC163是集成电路4位同步二进制计数器的一个例子。逻辑符号如图8.19所示，引脚编号在括号内。除了前面所讨论的一般的同步二进制计数器具有的基本功能之外，这种计数器还具有几个特征。

首先，通过把恰当的电平加到并行数据输入，计数器可以被同步预置到任意的4位二进制数。当低电平加到$\overline{LOAD}$（置数）输入时，计数器将在下一个时钟脉冲到来时读取数据输入的状态。因此，该计数器时序可以开始于任何4位二进制数。

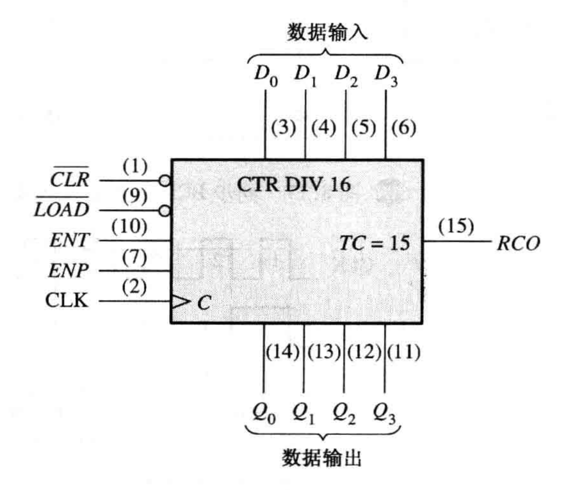


图8.19 74HC163 4位同步二进制计数器(限制符号CTR DIV 16表示具有16个状态的计数器)

同样，存在一个有效低电平清零输入$\overline{CLR}$，其同步复位计数器中所有的4个触发器。有两个使能输入*ENP*和*ENT*。这两个输入必须都为高电平，才能使计数器顺序经过它的二进制状态。只要有一个输入为低电平，计数器就不工作。当计数器到达序列15的最后一种状态时，异步(行波)时钟输出(*RCO*)就变为高电平，这称为终端计数(*TC* = 15)。这个输出(*RCO*)和使能输入(*ENP*和*ENT*)一起，使得计数器可以串接起来，从而得到更大的计数序列。

图8.20给出了计数器的时序图，开始时置位于12(1100)，然后进行加计数直到15(1111)。输入D_0为最低有效位，Q_0为最高有效位。

详细检查这个时序图。这将有助于解释本章的时序图或者生产商数据表上的时序图。作为开始，$\overline{CLR}$输入上的低电平脉冲使得所有的输出(Q_0、Q_1、Q_2和Q_3)变为低电平。

接下来，$\overline{LOAD}$输入上的低电平脉冲同步把数据输入(D_0、D_1、D_2和D_3)上的数据输入到计数器中。这些数据在$\overline{LOAD}$变为低电平之后的第一个时钟上升沿时，出现在Q输出上。这就是预置位的运算。在这个特殊的例子中，Q_0为低电平，Q_1为低电平，Q_2为高电平，Q_3为高电平。这是二进制数12(Q_0是最低有效位)。

计数器在接下来的三个上升时钟沿依次经过状态13、14和15。然后在接下来的时钟脉冲再循环回到0、1、2。注意*ENP*和*ENT*输入在状态时序期间都是高电平。当*ENP*为低电平时，计数器就会被禁止工作并且保持在二进制数2的状态。

8.2节 温故而知新

1. 一个同步计数器和一个异步计时器的区别是什么?
2. 解释计数器的预置位功能，例如74HC163。
3. 解释74HC163计数器的输入*ENP*、*ENT*和输出*RCO*的用途。

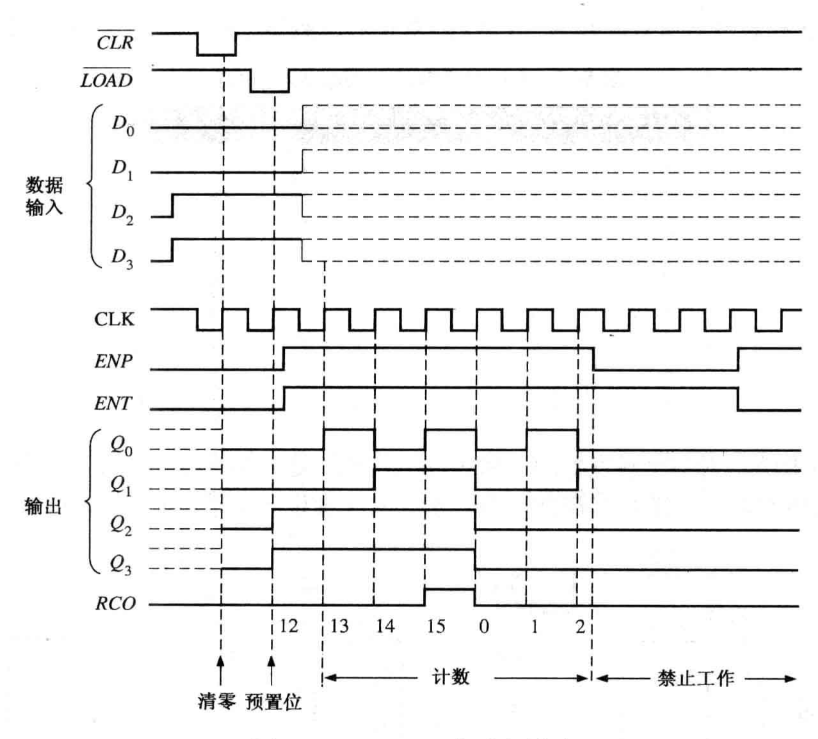


图 8.20 74HC163 的时序例子

8.3 加/减同步计数器

一般情况下，大多数加/减计数器都可以在序列中的任何地方反转。例如，3 位二进制计数器可以做到经过如下所示的序列：

$$\overbrace{0,1,2,3,4,5,}^{\text{加}}\underbrace{4,3,2,}_{\text{减}}\overbrace{3,4,5,6,7,}^{\text{加}}\underbrace{6,5,}_{\text{减}}\ \text{等等}$$

表 8.6 给出了一个 3 位二进制计数器的完整加/减时序。箭头表示计数器的加和减的两种运算模式下所进行的状态到状态的运动方向。检查加和减序列的 Q_0，指出 FF0 在每个时钟脉冲到来时的切换状态。因此，FF0 的 J_0 和 K_0 输入为

$$J_0 = K_0 = 1$$

对于加(UP)时序，Q_1 在 $Q_0 = 1$ 的下一个时钟脉冲到来时改变状态。对于减(DOWN)时序，Q_1 在 $Q_0 = 0$ 的下一个时钟脉冲到来时改变状态。因此，在下面的等式所表达的条件下，FF1 的 J_1 和 K_1 输入必须等于 1：

$$J_1 = K_1 = (Q_0 \cdot \text{UP}) + (\overline{Q}_0 \cdot \text{DOWN})$$

对于加时序来说，Q_2 在 $Q_0 = Q_1 = 1$ 的下一个时钟脉冲到来时改变状态。对于减时序来说，Q_2 在 $Q_0 = Q_1 = 0$ 的下一个时钟脉冲到来时改变状态。因此，在下面的等式所表达的条件下，FF2 的 J_2 和 K_2 输入必须等于 1：

$$J_2 = K_2 = (Q_0 \cdot Q_1 \cdot \text{UP}) + (\overline{Q}_0 \cdot \overline{Q}_1 \cdot \text{DOWN})$$

每个触发器的 J 和 K 输入所对应的每一个条件，都会在计数器序列中的恰当位置产生一个切换。

表 8.6　3 位二进制计数器的加/减时序

时钟脉冲	加	Q_2	Q_1	Q_0	减
0		0	0	0	
1		0	0	1	
2		0	1	0	
3		0	1	1	
4		1	0	0	
5		1	0	1	
6		1	1	0	
7		1	1	1	

如图 8.21 所示，对每个触发器的 J 和 K 输入使用上述逻辑等式，就是 3 位加/减二进制计数器的基本实现方法。注意 $UP/\overline{DOWN}$ 控制输入对于加是高电平，对于减是低电平。

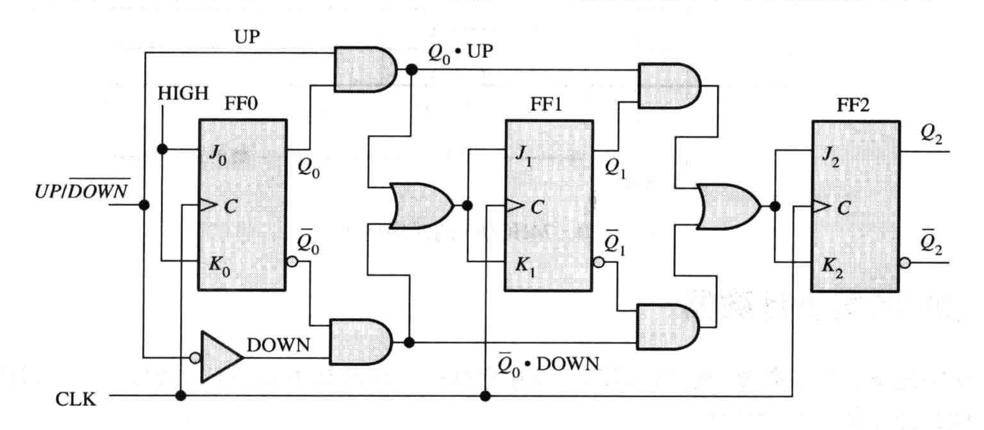


图 8.21　基本的 3 位加/减同步计数器。打开文件 F08-21 检验运行结果

例 8.4　如果时钟和 $UP/\overline{DOWN}$ 控制输入具有如图 8.22(a)所示的波形，给出时序图并确定 4 位同步二进制加/减计数器的时序。计数器开始于全 0 状态并且是上升沿触发。

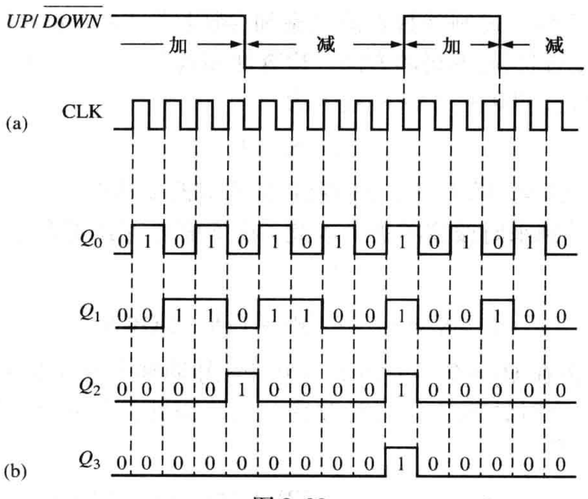


图 8.22

解：给出 Q 输出的时序图如图 8.22(a)所示。根据这些波形，计数器序列如表 8.7 所示。

表 8.7

Q_3	Q_2	Q_1	Q_0	
0	0	0	0	加
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	0	1	1	减
0	0	1	0	
0	0	0	1	
0	0	0	0	
1	1	1	1	
0	0	0	0	加
0	0	0	1	
0	0	1	0	
0	0	0	1	减
0	0	0	0	

相关问题：如果图 8.22(a)的 $UP/\overline{DOWN}$ 控制波形反相，请给出时序图。



74HC190 加/减十进制计数器

图 8.23 给出了 74HC190 的逻辑框图，它是集成电路加/减同步计数器的一个例子。计数的方向由加/减输入（$D/\overline{U}$）的电平确定，当这个输入为高电平时，计数器为减计数；当它是低电平时，计数器为加计数。同样，这个芯片可以被预置任意所需的 BCD 数字，当 $\overline{LOAD}$ 输入为低电平时，这个 BCD 数字由数据输入的状态来确定。

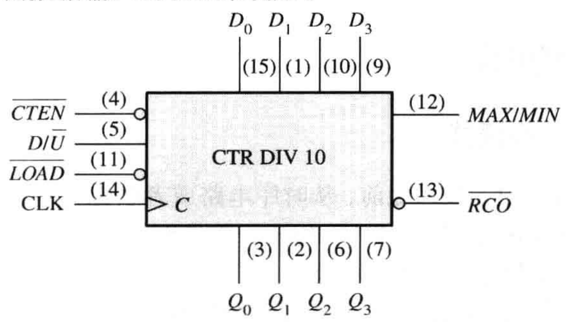


图 8.23 74HC190 加/减同步十进制计数器

当在加模式时到达终端计数值 9(1001)或者当在减模式时到达终端计数值 0(0000)，*MAX/MIN* 输出就会产生一个高电平脉冲。这个 *MAX/MIN* 输出、异步(行波)时钟输出（$\overline{RCO}$）及计数使能输入（$\overline{CTEN}$）一起，可用于级联计数器(级联计数器将在 8.5 节讨论)。

图 8.24 是一个时序图，给出了 74HC190 计数器预置为 7(0111)，然后经过一个加计数的时序并跟随一个减计数的时序。当计数器位于全 0 状态(*MIN*)或者 1001 状态(*MAX*)时，*MAX/MIN* 输出就是高电平。

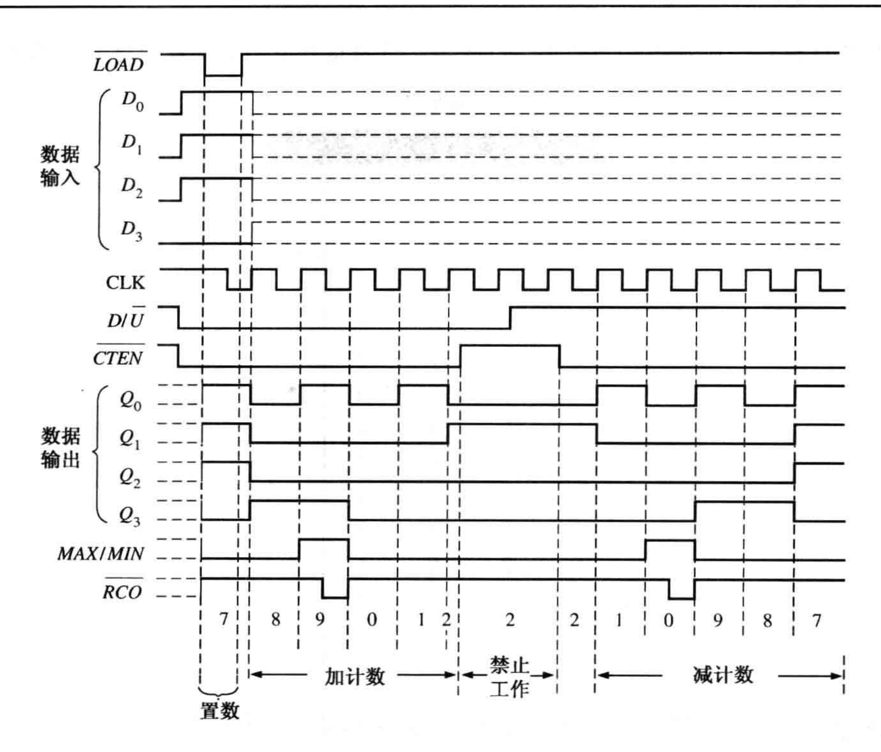


图 8.24　74HC190 的时序例子

8.3 节　温故而知新

1. 一个 4 位加/减计数器处于减模式，并且为 1010 状态。下一个时钟脉冲到来时，计数器的状态是什么?
2. 处在加模式的 4 位加/减计时器的最后一个状态是什么? 当处于减模式时，下一个状态是什么?

8.4　同步计数器的设计

8.4.1　时序电路的一般模式

在开始讨论特定计数器设计技术之前，从时序电路或者状态机的一般性定义开始：一般的时序电路由组合逻辑电路和存储器(触发器)组成，如图 8.25 所示。在一个时钟触发的时序电路中，存储器部分有一个时钟输入，如图所示。

电路的正常运算需要存储在存储器部分的信息和组合逻辑(I_0, I_1,…, I_m)的输入。在任何给定的时间，存储器所在的状态称为当前状态，并且将在时钟脉冲的作用下进入下一个状态，这个次态由激励线(Y_0, Y_1,…, Y_p)上的条件来确定。存储器的当前状态由状态变量(Q_0, Q_1,…, Q_x)来表示。这些状态变量和输入(I_0, I_1,…, I_m)一起确定了系统输出(O_0, O_1,…, O_n)。

并不是所有的时序电路都具有刚刚讨论的一般模式中所拥有的输入和输出变量。但是，所有的电路都具有激励变量和状态变量。计数器是时钟触发的时序电路的一个特例。在本节中，时序电路的一般设计步骤将在同步计数器的应用中一步步展开。

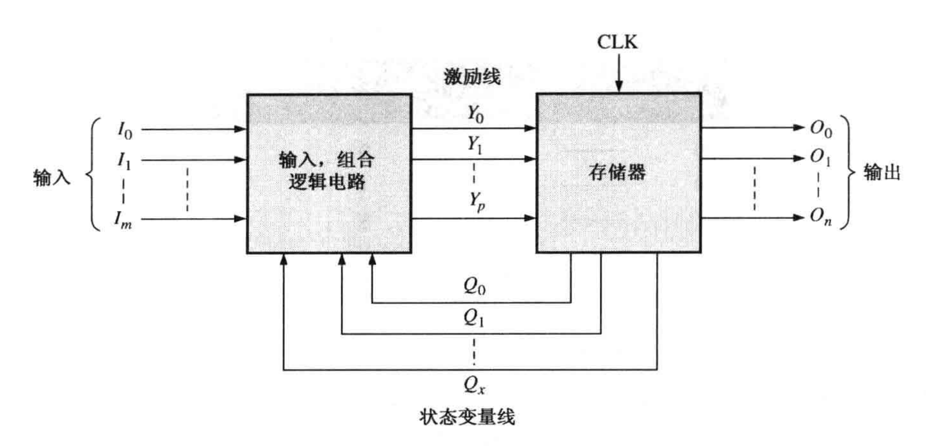


图 8.25 一般时钟触发的时序电路

8.4.2 步骤1：状态图

设计计数器的第一步是创建一个状态图。一个状态图给出了计数器在时钟的作用下状态行进的历程。作为一个例子，图8.26 给出了基本3位格雷码计数器的状态图。这个特殊电路除了时钟之外没有输入，除了计数器中每个触发器所具有的输出之外没有其他输出。这时，可以复习一下第2章有关格雷码的内容。

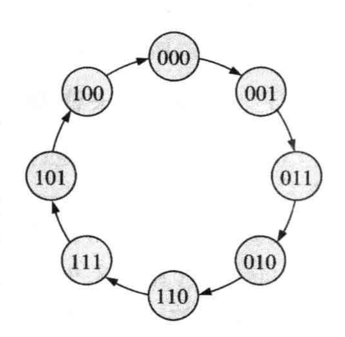


图 8.26 3位格雷码计数器的状态图

8.4.3 步骤2：次态表

一旦由状态图定义了时序电路，第二步就是推导出次态表，列出计数器的每一个状态(当前状态)和相应的下一个状态(次态)。次态就是计数器在时钟脉冲的作用下从当前状态转换过来的状态。次态表由状态图导出，如表8.8所示的3位格雷码计数器。Q_0是最低有效位。

表 8.8 3位格雷码计数器的次态表

当前状态			次态		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	0	0	1
0	0	1	0	1	1
0	1	1	0	1	0
0	1	0	1	1	0
1	1	0	1	1	1
1	1	1	1	0	1
1	0	1	1	0	0
1	0	0	0	0	0

8.4.4 步骤3：触发器转换表

表8.9是J-K触发器的状态转换表。通过给出触发器从当前状态到次态的Q输出，列出了所有可能的输出转换。Q_N是触发器的当前状态(在时钟脉冲之前)，而Q_{N+1}是次态(在时钟脉冲之后)。对于每个输出转换，列出了使得转换得以发生的输入J和K。X表示“无关”(输入可以为1或者0)。

表 8.9 J-K 触发器的转换表

输出转换			触发器输入	
Q_N		Q_{N+1}	J	K
0	⟶	0	0	X
0	⟶	1	1	X
1	⟶	0	X	1
1	⟶	1	X	0

Q_N: 当前状态　Q_{N+1}: 次态　X: “无关”

为了设计此计数器，基于次态表(见表 8.8)的转换表应用于计数器中的每一个触发器。例如，对于当前状态 0000，Q_0从当前状态 0 到次状态 1。为了使这种情况得以发生，J_0必须为 1，不要考虑 K_0是什么($J_0=1$，$K_0=X$)，正如在转换表(见表 8.9)中看到的那样。接下来，Q_1的当前状态为 0 并且在下一个状态保持为 0。对于这个转换，$J_1=0$，$K_0=X$。最后，Q_2的当前状态为 0 而在下一个状态保持为 0。所以，$J_2=0$ 和 $K_2=X$。对于表 8.8 中的每一个当前状态，重复这样的分析。

8.4.5 步骤 4：卡诺图

卡诺图可以用来确定计数器中每个触发器的 J 和 K 输入所需要的逻辑。对于每个触发器的 J 输入有一个卡诺图，每一个 K 输入也有一个卡诺图。在这个设计过程中，卡诺图中的每个小方格都表示列在表 8.8 的计数器时序中的一个当前状态。

由转换表(见表 8.9)中的 J 和 K 状态，1、0 或者 X 置于卡诺图上每一个当前状态小方格中，放什么取决于特定触发器 Q 输出的转换。为了解释这个过程，图 8.27 中给出了两个示例输入项，即最低有效位触发器 Q_0的 J_0和 K_0输入。

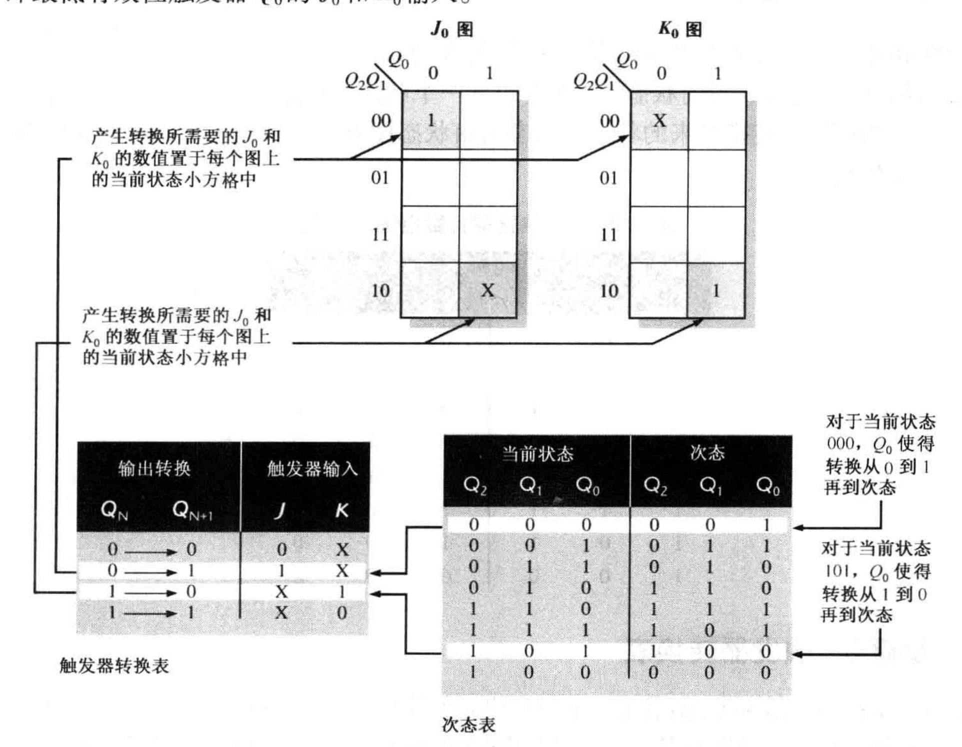


图 8.27 表 8.8 和表 8.9 所表示的计数器时序映射过程的例子

计数器中所有三个触发器的完整卡诺图如图8.28所示。小方格的分组如图所示，并且推导了和每组相对应的布尔表达式。

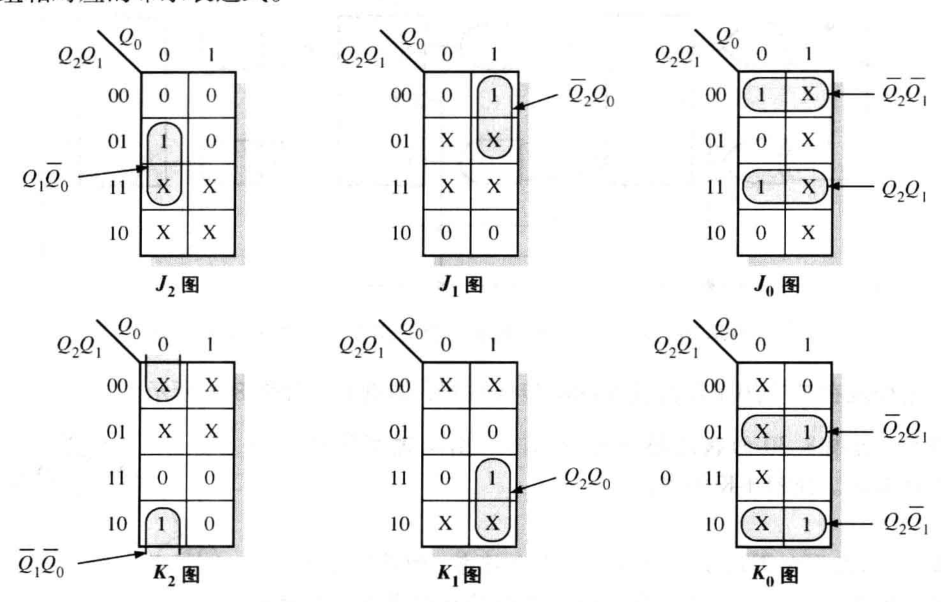


图8.28　当前状态 J 和 K 输入的卡诺图

8.4.6　步骤5：触发器输入的逻辑表达式

从图8.28的卡诺图中，得到如下所示的每个触发器的 J 和 K 输入的表达式：

$$J_0 = Q_2Q_1 + \overline{Q}_2\overline{Q}_1 = \overline{Q_2 \oplus Q_1}$$

$$K_0 = Q_2\overline{Q}_1 + \overline{Q}_2Q_1 = Q_2 \oplus Q_1$$

$$J_1 = \overline{Q}_2Q_0$$

$$K_1 = Q_2Q_0$$

$$J_2 = Q_1\overline{Q}_0$$

$$K_2 = \overline{Q}_1\overline{Q}_0$$

8.4.7　步骤6：计数器的实现

最后一步是从 J 和 K 输入表达式中实现组合逻辑，并连接触发器来构成完整的3位格雷码计数器，如图8.29所示。

设计计数器所需步骤的总结如下所示。一般情况下，这些步骤可以应用于任何时序电路。

1. 确定计数器时序并绘制状态图。
2. 从状态图推导出次态表。
3. 开发转换表，给出每个转换所需要的触发器输入。转换表对于给定触发器的类型总是相同的。
4. 把 J 和 K 状态从转换表转移到卡诺图上。每个触发器的每个输入都有一个卡诺图。
5. 把卡诺图小方格分组，以产生并导出每个触发器输入的逻辑表达式。
6. 使用组合逻辑实现表达式，并且连接触发器以创建计数器。

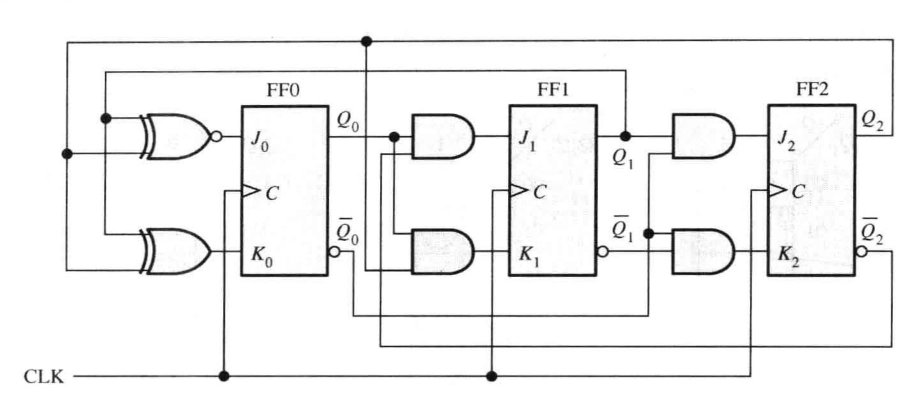


图 8.29　3 位格雷码计数器。打开文件 F08-29 检验该操作

这个过程现在被应用于设计其他的同步计数器，如例 8.5 和例 8.6 所示。

例 8.5　为图 8.30 的状态图中所示的不规则二进制计数时序设计一个计数器。使用 J-K 触发器。

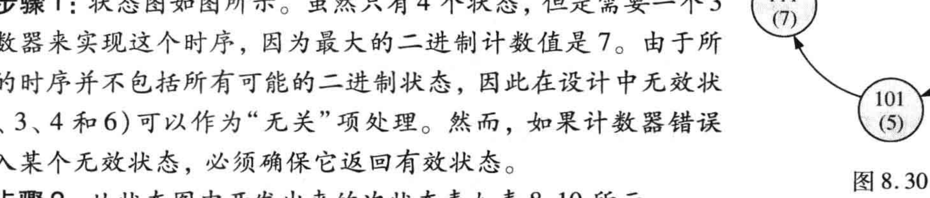


图 8.30

解：

步骤 1： 状态图如图所示。虽然只有 4 个状态，但是需要一个 3 位计数器来实现这个时序，因为最大的二进制计数值是 7。由于所需要的时序并不包括所有可能的二进制状态，因此在设计中无效状态(0、3、4 和 6)可以作为“无关”项处理。然而，如果计数器错误地进入某个无效状态，必须确保它返回有效状态。

步骤 2： 从状态图中开发出来的次状态表如表 8.10 所示。

步骤 3： J-K 触发器的转换表列于表 8.11 中。

表 8.10　次态表

当前状态			次态		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	1	0	1	0
0	1	0	1	0	1
1	0	1	1	1	1
1	1	1	0	0	1

表 8.11　J-K 触发器的转换表

输出转换			触发器输入	
Q_N		Q_{N+1}	J	K
0	⟶	0	0	X
0	⟶	1	1	X
1	⟶	0	X	1
1	⟶	1	X	0

步骤 4： J 和 K 输入绘制在图 8.31 中的当前状态卡诺图上。同样，“无关”项可以放置在相应的小方格中，这些小方格对应无效状态 000、011、100 和 110，由 X 表示。

步骤 5： 将 1 分组，尽可能利用较多的“无关”项状态来实现最大的简化，如图 8.31 所示。注意当图中所有的小方格都被分组时，表达式就简单地等于 1。从图中得到每个 J 和 K 输入的表达式如下：

$$J_0 = 1, K_0 = \overline{Q}_2$$
$$J_1 = K_1 = 1$$
$$J_2 = K_2 = Q_1$$

步骤 6： 计数器的实现如图 8.32 所示。

分析显示，如果该计数器偶尔进入无效状态(0、3、4、6)中的某一个状态，它总是能够返回一个有效状态，根据下面的序列：0→3→4→7，6→1。

Q_2Q_1 \ Q_0	0	1
00	X	0
01	1	X
11	X	X
10	X	X

J_2图（圈：Q_1）

Q_2Q_1 \ Q_0	0	1
00	X	1
01	X	X
11	X	X
10	X	1

J_1图（圈：1）

Q_2Q_1 \ Q_0	0	1
00	X	X
01	1	X
11	X	X
10	X	X

J_0图（圈：1）

Q_2Q_1 \ Q_0	0	1
00	X	X
01	X	X
11	X	1
10	X	0

K_2图（圈：Q_1）

Q_2Q_1 \ Q_0	0	1
00	X	X
01	1	X
11	X	1
10	X	X

K_1图（圈：1）

Q_2Q_1 \ Q_0	0	1
00	X	1
01	X	X
11	X	0
10	X	0

K_0图（圈：$\overline{Q}_2$）

图 8.31

图 8.32

相关问题：对这个分析，即计数器总是能够从无效状态返回(最终)有效状态进行检验。

例 8.6　开发一个具有格雷码时序的3位加/减计数器。当 $UP/\overline{DOWN}$ 控制输入为1时，该计数器应当加计数，而当控制输入为0时，应当减计数。

解：

步骤1：状态图如图8.33所示。每个箭头旁边的1和0表示 $UP/\overline{DOWN}$ 控制输入 Y 的状态。

步骤2：从状态图中所导出的次态表如表8.12所示。注意对于每一个当前状态都有两个可能的次态，这取决于 $UP/\overline{DOWN}$ 控制变量 Y。

步骤3：J-K触发器的转换表重复于表8.13中。

步骤4：触发器 J 和 K 输入的卡诺图如图8.34所示。$UP/\overline{DOWN}$ 控制输入 Y 和 Q_0、Q_1、Q_2 都被考虑为状

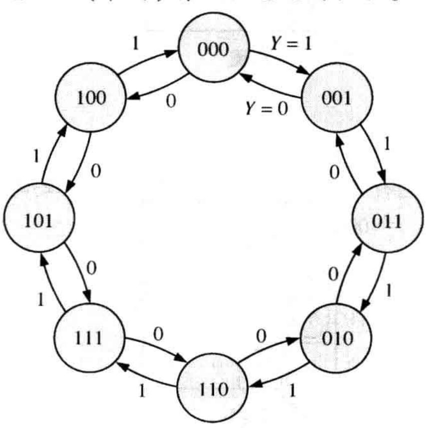


图 8.33　3位加/减格雷码计数器的状态图

态变量。使用次态表，将表 8.13“触发器输入”一列中的信息转移到卡诺图上，以表示计数器的每一个当前状态。

表 8.12　3 位加/减格雷码计数器的次态表

当前状态			次态 Y = 0（减）			次态 Y = 1（加）		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	Q_2	Q_1	Q_0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	1
0	1	1	0	0	1	0	1	0
0	1	0	0	1	1	1	1	0
1	1	0	0	1	0	1	1	1
1	1	1	1	1	0	1	0	1
1	0	1	1	1	1	1	0	0
1	0	0	1	0	1	0	0	0

$Y = UP/\overline{DOWN}$控制输入。

表 8.13　J-K 触发器的转换表

输出转换 Q_N		Q_{N+1}	触发器输入 J	K
0	⟶	0	0	X
0	⟶	1	1	X
1	⟶	0	X	1
1	⟶	1	X	0

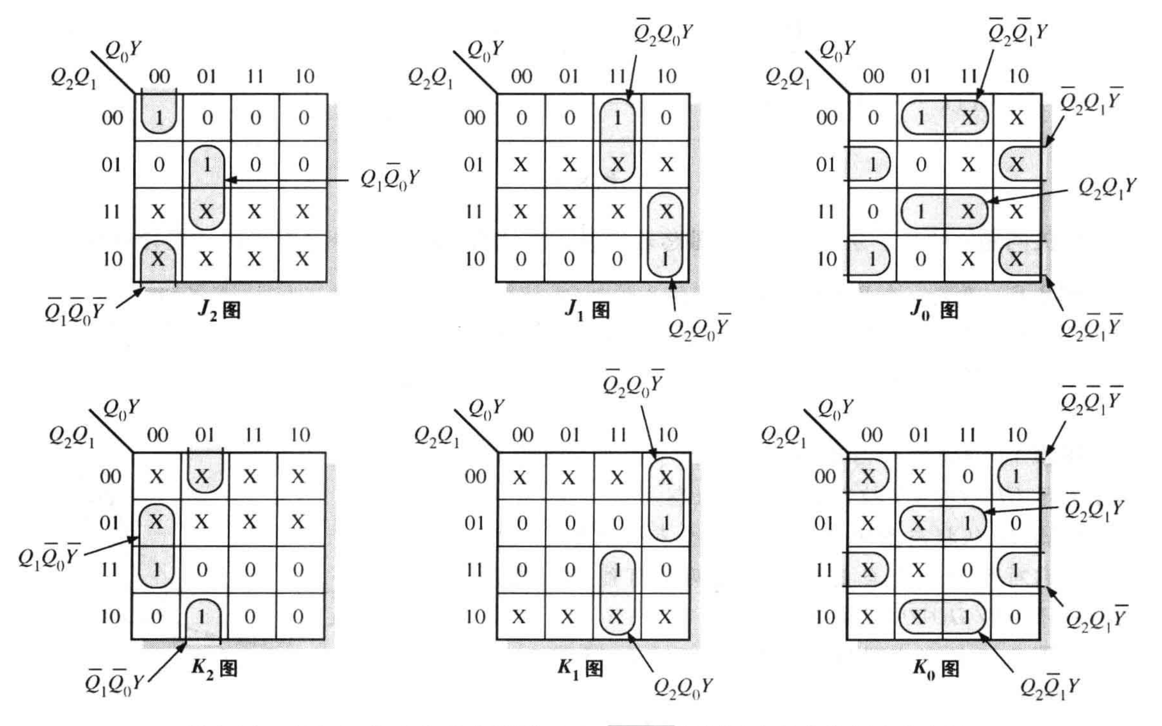


图 8.34　表 8.12 的 J 和 K 卡诺图。$UP/\overline{DOWN}$ 控制输入 Y 作为第 4 个变量

步骤5：把1组合成尽可能大的组，只要可能就使用“无关”项。从这些组中分解出变量，J 和 K 输入的表达式如下所示：

$$J_0 = Q_2Q_1Y + Q_2\overline{Q}_1\overline{Y} + \overline{Q}_2\overline{Q}_1Y + \overline{Q}_2Q_1\overline{Y} \qquad K_0 = \overline{Q}_2\overline{Q}_1\overline{Y} + \overline{Q}_2Q_1Y + Q_2\overline{Q}_1Y + Q_2Q_1\overline{Y}$$

$$J_1 = \overline{Q}_2Q_0Y + Q_2Q_0\overline{Y} \qquad K_1 = \overline{Q}_2Q_0\overline{Y} + Q_2Q_0Y$$

$$J_2 = Q_1\overline{Q}_0Y + \overline{Q}_1\overline{Q}_0\overline{Y} \qquad K_2 = Q_1\overline{Q}_0\overline{Y} + \overline{Q}_1\overline{Q}_0Y$$

步骤6：使用组合逻辑实现 J 和 K 的等式，完整的计数器如图8.35所示。

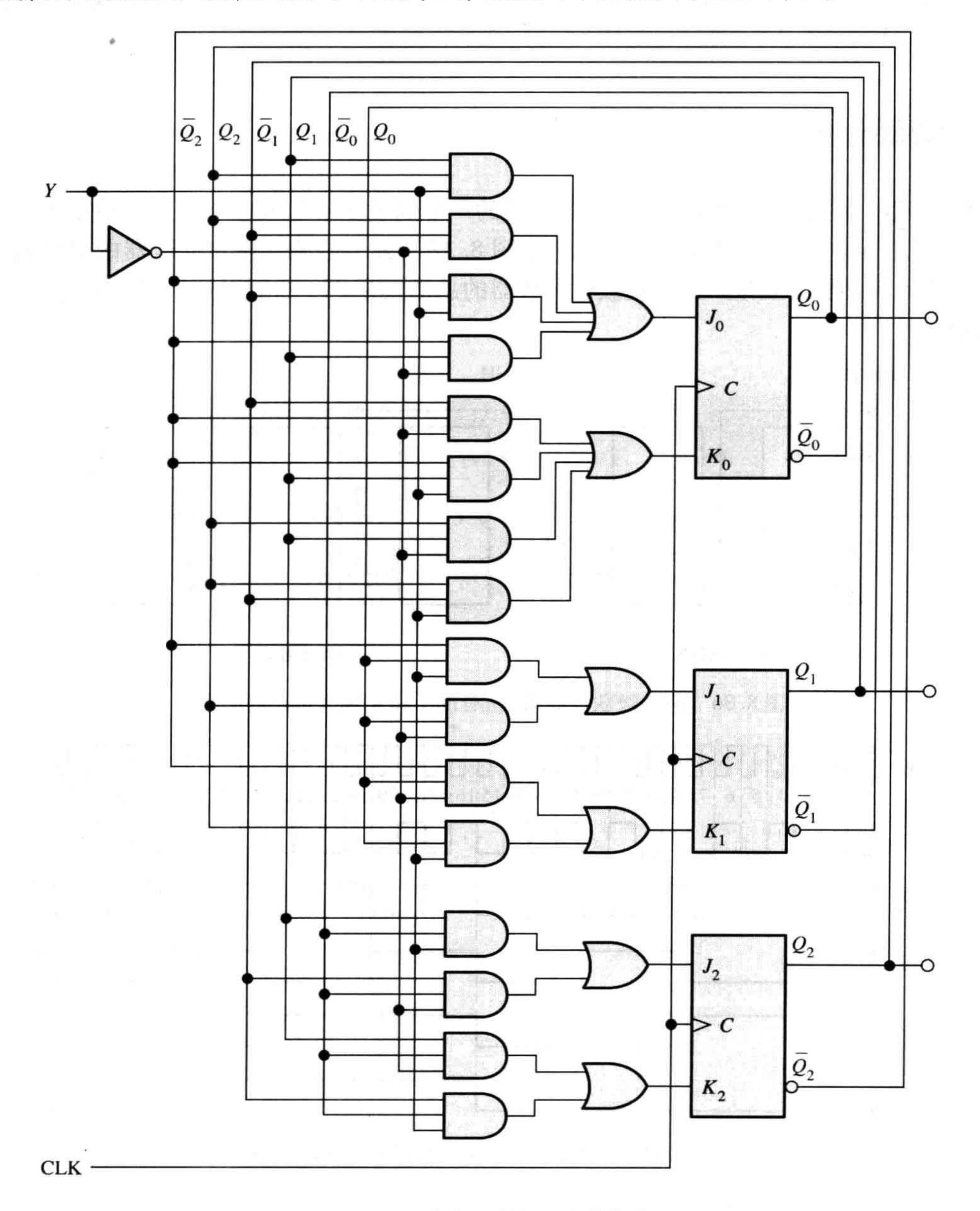


图8.35 3位加/减格雷码计数器

相关问题：验证图8.35中的逻辑与步骤5中的表达式相符。

8.4 节 温故而知新

1. 一个 J-K 触发器的当前状态为复位，下一个状态默许为置位，那么如何设置 J 和 K?
2. 一个 J-K 触发器的当前状态为置位，下一个状态默许为置位，那么如何设置 J 和 K?
3. 一个二进制触发器的当前状态是 $Q_3\bar{Q}_2Q_1\bar{Q}_0 = 1010$。
   (a) 下一个状态是什么?
   (b) 在时钟的作用下，保证触发器进入合适的下一个状态，那么每个触发器必须处在什么情况下?

8.5 级联计数器

异步级联 两个计数器连接为级联计数器的一个例子如图 8.36 所示，图中给出了一个 2 位和一个 3 位的异步(行波)计数器。时序图如图 8.37 所示。注意，模 8 计数器的最终输出 Q_4，每 32 个输入时钟脉冲发生一次。此级联计数器的总的模是 32；也就是说，它们表现为 32 分频的计数器。

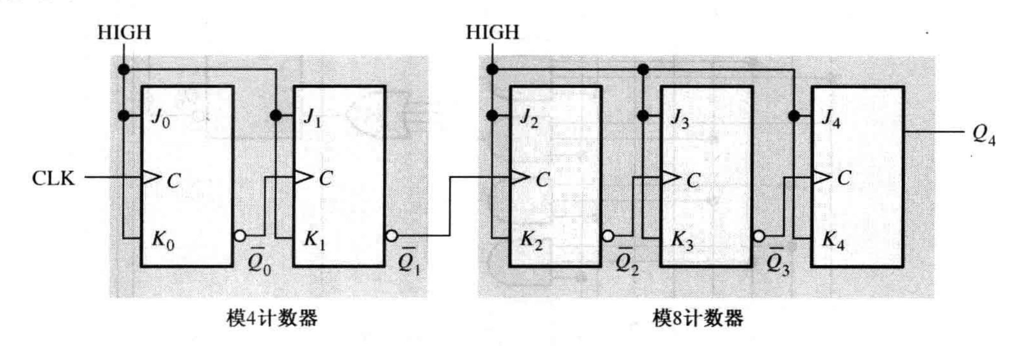


图 8.36 两个级联计数器(所有的 J 和 K 输入都是高电平)

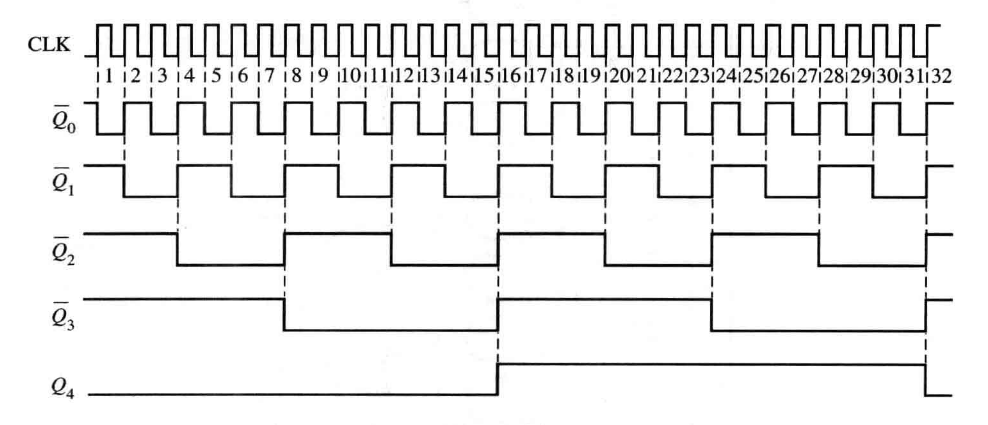


图 8.37 图 8.36 的级联计数器配置的时序图

◇ 级联计数器的总的模等于单个模的积。

同步级联 在级联配置中运行同步计数器时，需要使用计数使能和终端计数功能来实现更高模运算。在一些芯片上，计数使能被简单地标记为 $CTEN$(或者一些诸如 G 之类的其他名称)，终端计数(TC)和一些 IC 计数器上的异步(行波)时钟输出(RCO)比较相似。

图8.38给出了两个十进制计数器的级联。计数器1的终端计数(TC)输出连接到计数器2的计数使能($CTEN$)输入上。计数器2被它的$CTEN$输入上的低电平禁止，直至计数器1到达它的最后或者终端状态，并且其终端计数输出变为高电平。这个高电平现在使得计数器2工作，结果当计数器1到达它的终端计数(CLK10)之后的第一个时钟脉冲到来时，计数器2从它的初始状态变到第二个状态。在计数器1完成整个第二次循环之后(当计数器1第二次到达终端计数时)，使得计数器2再次工作并进入下一个状态。这个时序继续进行。由于它们是十进制计数器，所以计数器1在计数器2完成它的第一次循环之前，必须经过10个完整的循环。也就是说，对应于计数器1的每10个循环，计数器2经过一个循环。因此，计数器2将在100个时钟脉冲之后完成一个循环，这两个级联计数器的总的模是10×10=100。

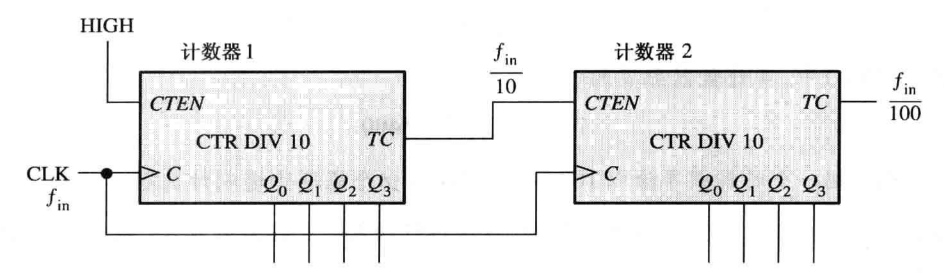


图8.38　使用两个级联十进制计数器的模100计数器



计算机小知识

上一次提到的时间戳计数器(TSC)是一个64位计数器。如果此计数器(或者任何全模64位计数器)在1 GHz的时钟频率下工作，观察起来会很有趣，它将耗费583年的时间来经过它所有的状态，并到达它的终端计数。与此形成对照的是，32位全模计数器在1 GHz的时钟频率下工作时，只需4.3秒左右的时间就可以经过所有的状态。

当视为分频器时，图8.38中的电路就会把输入时钟频率进行100分频。级联计数器常常用来对高频率时钟信号进行分频，从而得到精确度很高的脉冲频率。用于此目的的级联计数器配置有时称为递减计数链(countdown chain)。例如，假设有一个1 MHz的基本时钟频率，想要得到100 kHz、10 kHz、1 kHz的频率，可以使用一系列级联的十进制计数器。如果1 MHz信号被一分为10，输出就是100 kHz。如果100 kHz信号再被一分为10，输出就是10 kHz。再次除以10将会达到1 kHz的频率。这个递减计数链的一般实现方法如图8.39所示。

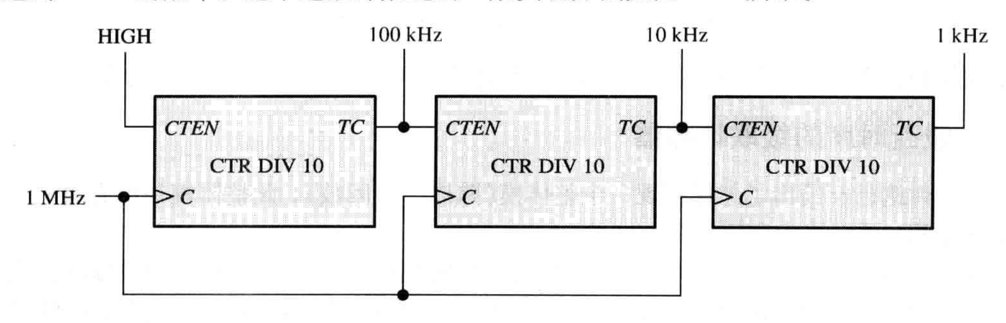


图8.39　3个级联十进制计数器构成1000分频，具有中间10分频和100分频输出

例 8.7 确定如图 8.40 所示的两个级联计数器配置的整体模。

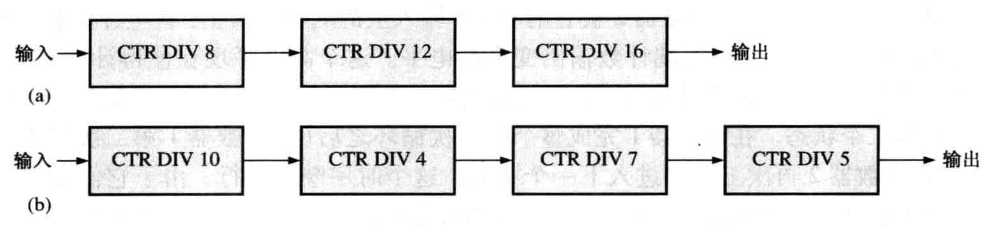


图 8.40

解: 在图 8.40(a)中,3 计数器配置的整体模是

$$8 \times 12 \times 16 = \mathbf{1536}$$

在图 8.40(b)中,4 计数器配置的整体模是

$$10 \times 4 \times 7 \times 5 = \mathbf{1400}$$

相关问题: 把一个时钟频率除以 100 000,需要多少个级联十进制计数器?

例 8.8 如连接框图,使用 74HC190 加/减译码计数器,连接为加模式,从 1 MHz 时钟获得一个 10 kHz 的时钟。

解: 为了从 1 MHz 时钟中得到 10 kHz,需要除以因数 100。两个 74HC190 计数器必须依照图 8.41 所示的那样进行级联。左边的计数器每 10 个时钟脉冲产生一个终端计数输出(MAX/MIN)。右边的计数器每 100 个时钟脉冲产生一个终端计数输出(MAX/MIN)。

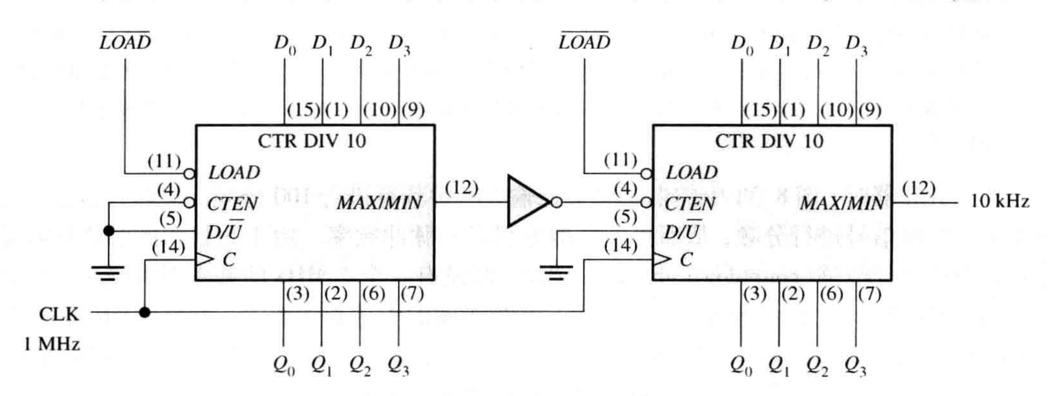


图 8.41 使用两个 74HC190 十进制计数器的 100 分频计数器

相关问题: 确定图 8.41 中第二个计数器(右边的计数器)的 Q_0 输出上波形的频率。

8.5.1 截断时序的级联计数器

前面的讨论已经给出了怎样实现一个整体模(除以一个因数),此整体模是所有级联计数器独立模的乘积,可以称为全模级联。

某些应用需要的整体模常常小于由全模级联所实现的整体模。也就是说,级联计数器必须实现截断时序。为了阐释这个方法,使用图 8.42 中的级联计数器配置。这个特殊的电路使用了 4 个 74HC161 4 位同步二进制计数器。如果这 4 个计数器(总共 16 位)以全模方式级联,模应当是

$$2^{16} = 65\ 536$$

假设某个特定的应用需要一个40 000分频的计数器(模40 000)。65 536和40 000的差值为25 536，也就是必须从全模时序中删除的状态个数。图8.42的电路所使用的技术就是，每循环一次都将级联计数器预置到25 536(十六进制为63C0)，所以在每个完整的循环中它将从25 536计数到65 535。所以计数器的每个完整的循环都由40 000个状态组成。

注意在图8.42中，最右边的计数器的*RCO*输出是反相的，并加在每个4位计数器的$\overline{LOAD}$输入上。计数器每次到达它的终端计数65 535，也就是1111111111111111_2时，*RCO*就会变为高电平，并使得并行数据输入($63C0_{16}$)在同步时钟脉冲的作用下进入计数器。因此，每40 000个时钟脉冲，最右边的4位计数器只有一个*RCO*脉冲输出。使用这种技术，通过将计数器在每次循环中都同步置入合适的初始状态，就可以实现任何模的计数器。

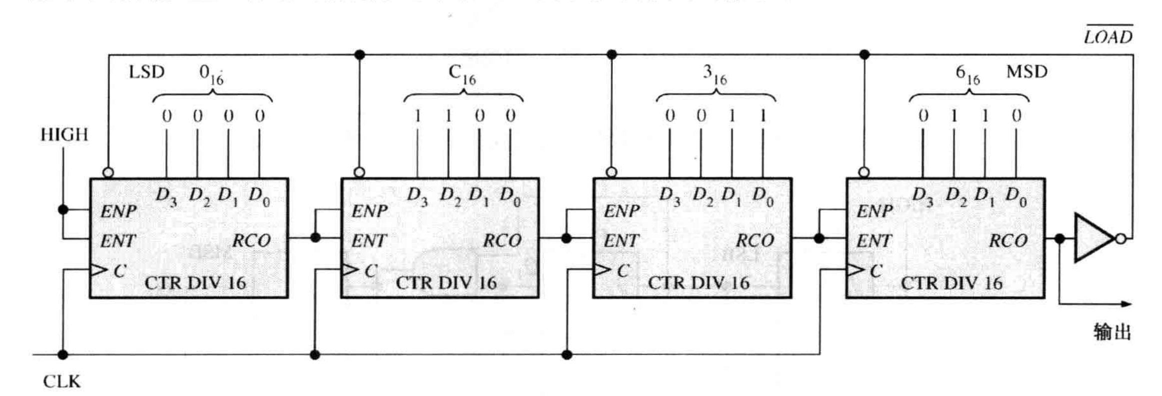


图8.42 使用74HC161 4位二进制计数器的40 000分频计数器。注意每个并行的数据输入都以二进制顺序给出(最右边的位D_0是每个计数器中的最低有效位)

8.5节 温故而知新

1. 完成一个1000分频的计数器需要多少个十进制计数器？那么10 000分频呢？
2. 给出如何获取如下分频的一般框图，使用触发器、十进制计数器、4位二进制计数器或它们的组合：
   (a)20分频计数器 (b)32分频计数器 (c)160分频计数器 (d)320分频计数器

8.6 计数器译码

假设希望译码一个3位二进制计数器的状态6(110)。当$Q_2=1$、$Q_1=1$和$Q_0=0$时，高电平就会出现在译码门的输出上，表示此计数器处于状态6。实现方法如图8.43所示。这称为高电平有效译码。用与非门取代与门，以提供低电平有效译码。

例8.9 实现一个3位同步计数器二进制状态2和二进制状态7的译码。给出整个计数器时序图和译码门的输出波形。二进制$2=\overline{Q}_2Q_1\overline{Q}_0$，二进制$7=Q_2Q_1Q_0$。

解： 参见图8.44。该3位计数器原来已经在8.2节讨论过(见图8.14)。

相关问题： 给出3位计数器中状态5的译码逻辑。

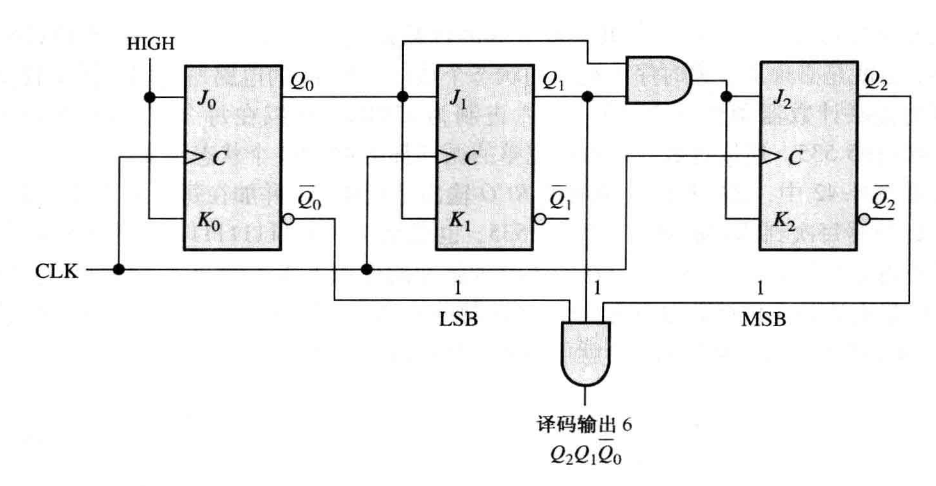


图 8.43　状态 6(110)的译码。打开文件 F08-43 检验运行结果

图 8.44　高电平有效的 2 和 7 译码的 3 位计数器。打开文件 F08-44 检验运行结果

8.6.1 译码假信号

在第6章介绍了译码过程产生的假信号问题。正如所看到的那样，由于异步计数器中的异步(行波)效应所产生的传输延迟，使得异步计数器输出变化而产生的状态转换在时间上稍有差别。这些状态转换使得连接计数器的译码器输出上，产生短期的不希望得到的电压窄脉冲(假信号)。对于同步计数器来说，也可能发生某种程度的假信号问题，因为计数器中从时钟到每个触发器的 Q 输出的时间延迟可能有很小的差别。

◇ 假信号就是不希望得到的电压窄脉冲。

图8.45给出了一个基本异步BCD十进制计数器，它连接到一个BCD-十进制译码器上。为了观察在此情况下会发生什么，我们来看时序图，其中也考虑了传输延迟，如图8.46所示。注意这些延迟产生了短期的错误状态。每个临界转换处的错误二进制状态值表示在图上。在译码器输出上可以看到假信号结果。

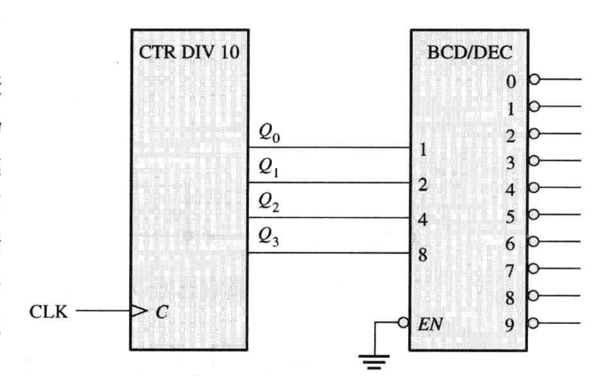


图8.45 基本十进制(BCD)计数器和译码器

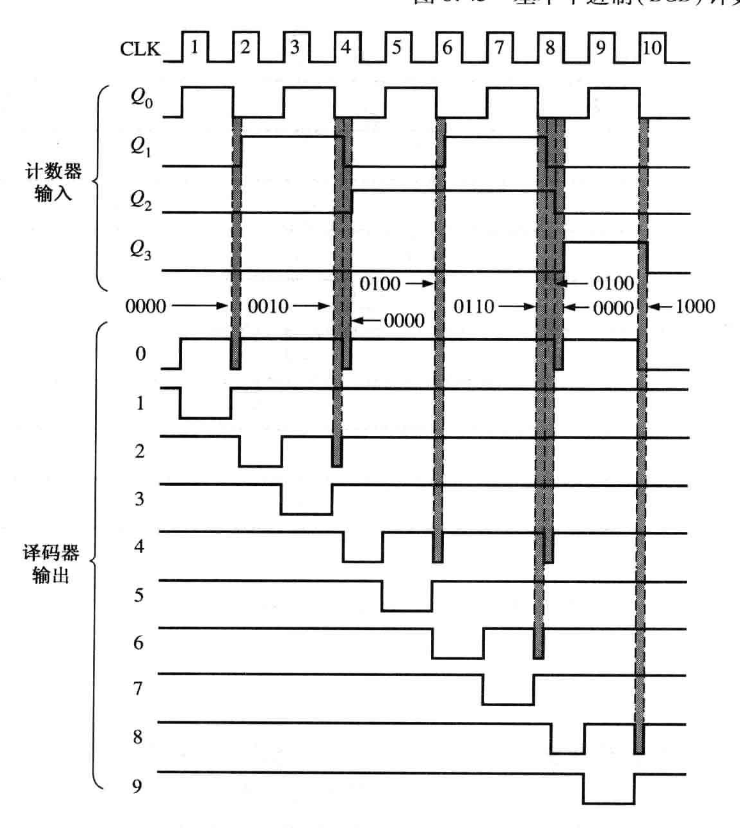


图8.46 图8.45中的译码器具有假信号的输出。为了解释方便，图中将假信号宽度夸大了，实际上假信号通常只有几纳秒宽

一种消除假信号的方法是在假信号消失以后才使译码能够输出。这种方法称为选通(strobing),可以在这样的情况下实现,即使用高电平有效时钟(对计数器而言)的低电平来使译码器工作,如图 8.47 所示。最后改善的时序图如图 8.48 所示。

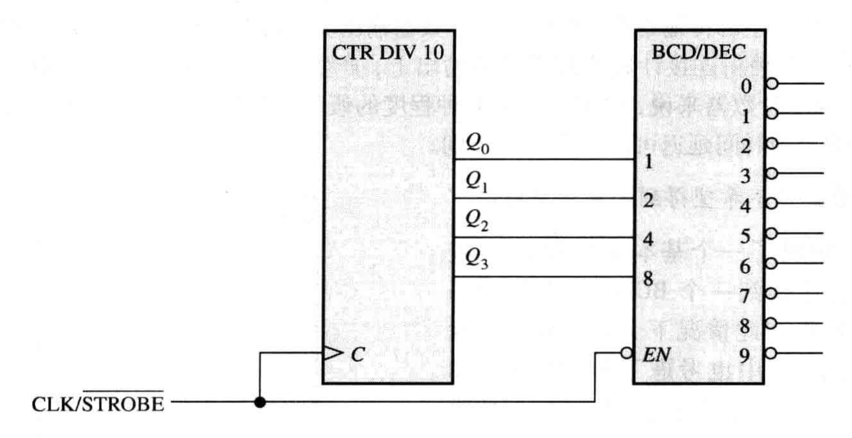


图 8.47　具有选通以消除假信号的基本十进制计数器和译码器

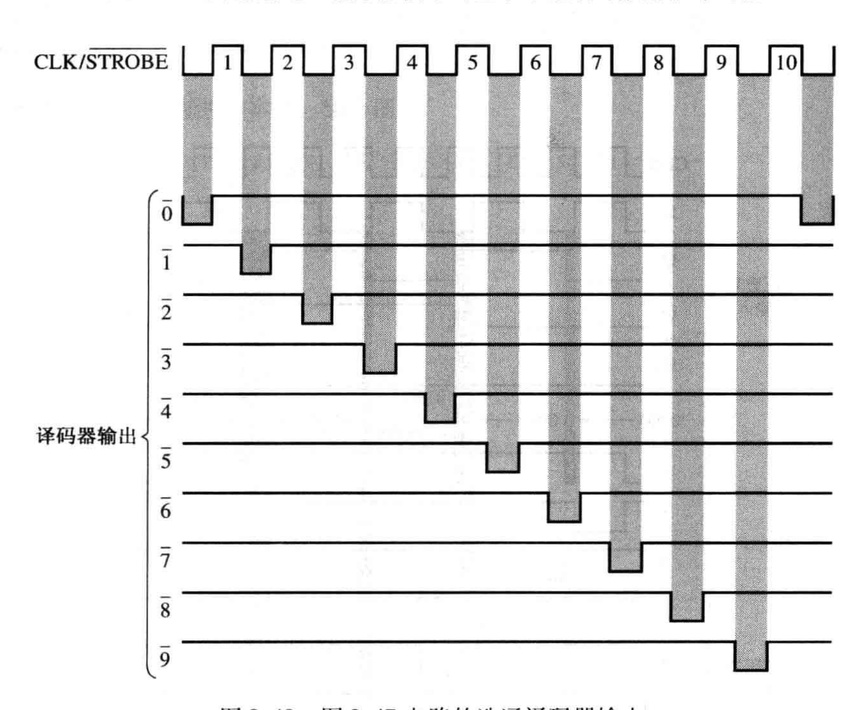


图 8.48　图 8.47 电路的选通译码器输出

8.6 节　温故而知新

1. 当 4 位二进制异步计数器状态变化如下时,可能发生的瞬间状态是什么?

(a)计数 2 到计数 3　　(b)计数 3 到计数 4

(c)计数 10_{10} 到计数 11_{10}　　(d)计数 15 到计数 0

8.7 计数器应用

8.7.1 数字时钟

计数器应用的一个常见例子是计时系统。图8.49是显示秒、分、小时的数字时钟的简化逻辑框图。首先，60 Hz的正弦交流电压变换为60 Hz的脉冲波形，并且由一个60分频的计数器分频为1 Hz的脉冲波形，这个计数器由一个10分频计数器紧接着一个6分频的计数器组成。秒和分计数也是由60分频的计数器产生的，其细节如图8.50所示。这些计数器从0计数到59然后再循环回到0；同步十进制计数器用在了这个特殊实现方法中。注意6分频的部分由一个具有截断时序的十进制计数器组成，截断时序通过使用译码器计数6来异步清零计数器而实现。终端计数59，也被译码以使计数器链中的下一个计数器工作。

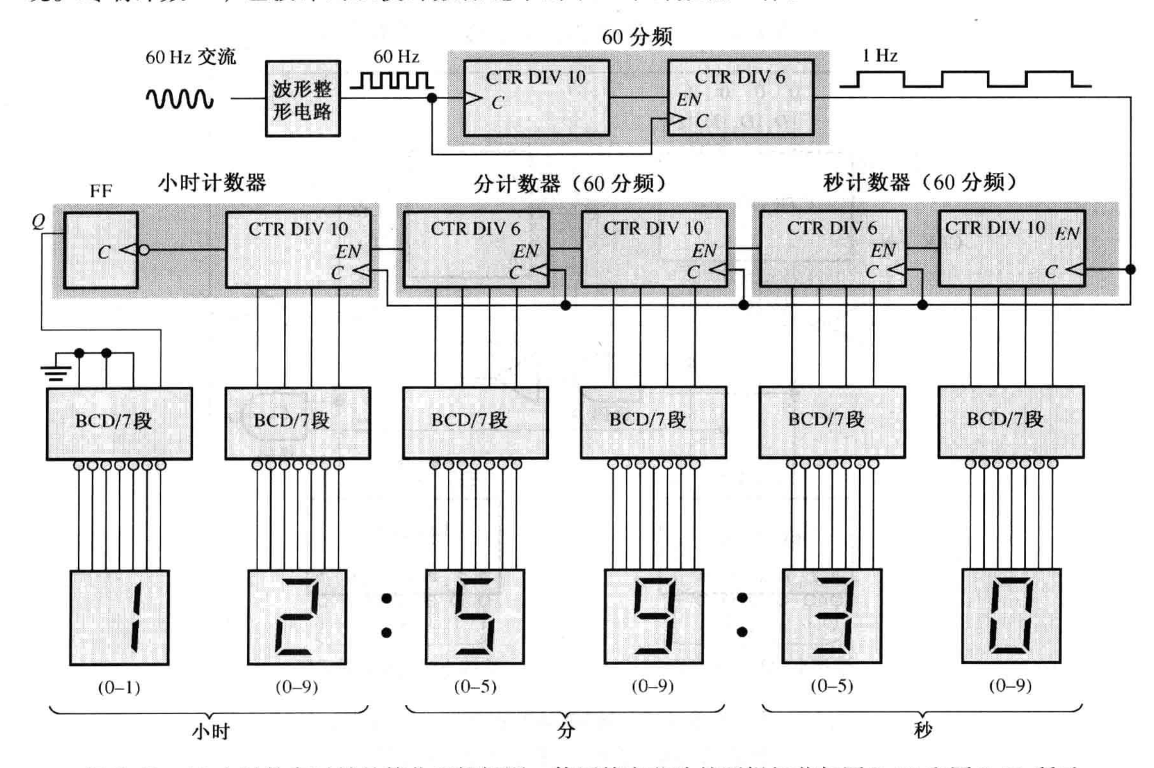


图8.49 12小时数字时钟的简化逻辑框图。使用特定芯片的逻辑细节如图8.50和图8.51所示

小时计数器由一个十进制计数器和一个触发器组成，如图8.51所示。考虑到初始时两个十进制计数器和触发器都处于复位状态，并且译码12门和译码9门输出都是高电平。十进制计数器顺序经过0~9的所有状态，并且在时钟脉冲到来时，从9再循环回到0，触发器进入置位状态($J=1$，$K=0$)。这样就点亮了十位数小时显示器上的1。总计数现在为10(十进制计数器处于0状态和触发器为置位状态)。

接下来，总计数顺序计到11然后再到12。在状态12时，十进制计数器的Q_2输出为高电平，触发器仍然为置位，因此译码12门输出为低电平。这就使十进制计数器的$\overline{LOAD}$输入有效。在下一个时钟脉冲到来时，十进制计数器由数据输入预置到状态0001，而触发器处于复位($J=0$，$K=1$)。如同所见，这样的逻辑使得计数器从12回到1，而不是0。

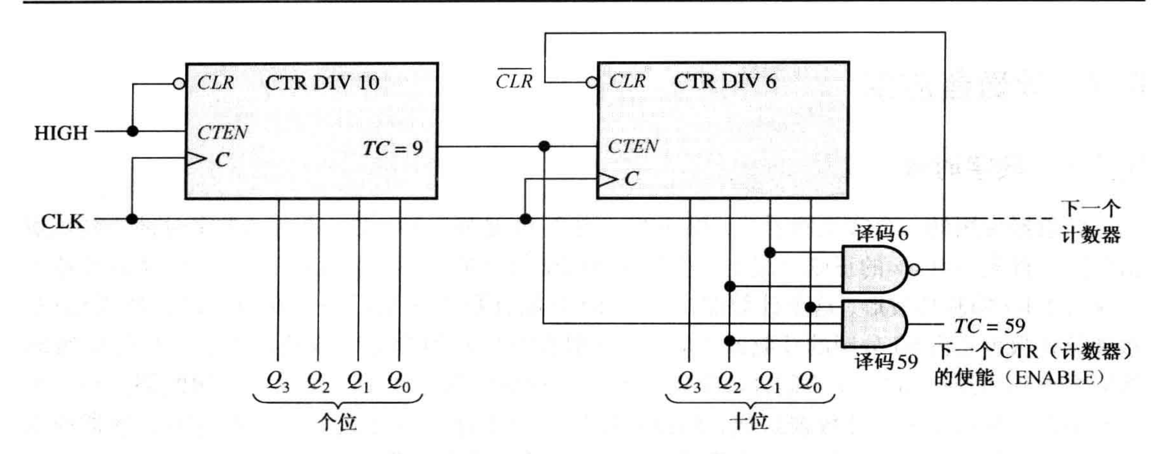


图 8.50 使用同步十进制计数器的典型 60 分频计数器的逻辑框图。注意输出处于二进制顺序(最右边的位为最低有效位)

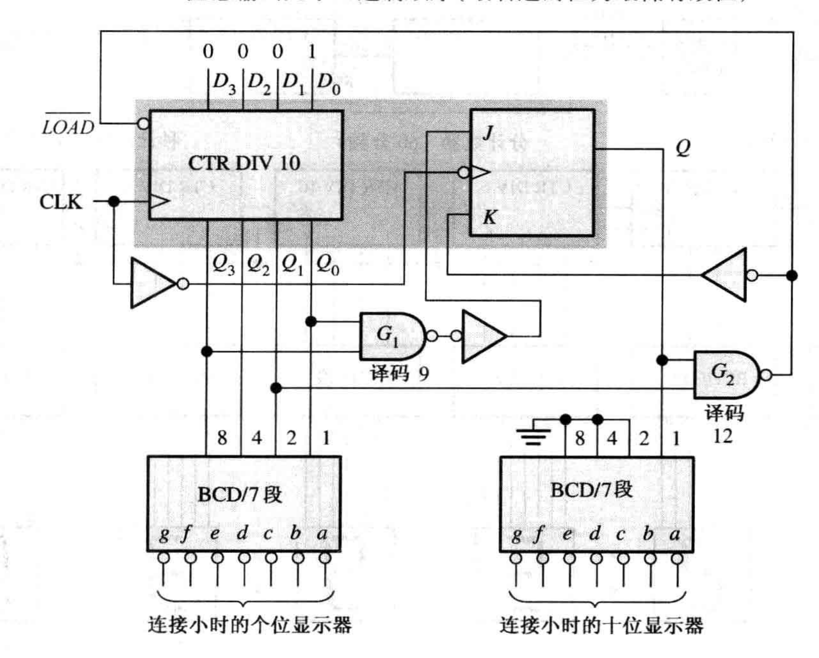


图 8.51 小时计数器和译码器的逻辑框图。注意在计数器输入和输出上,最右边的位是最低有效位

8.7.2 停车控制

这个计数器例子阐述了使用加/减计数器来解决日常问题。这个问题是设计一种检测手段，检测拥有 100 个车位的停车库中的可用车位，并且通过点亮显示信号以提供停满条件的指示，同时降下入口处的起落杆。

解决此问题的系统由以下几部分构成：(1)车库入口和出口处的光电传感器；(2)一个加/减计数器和相关电路；(3)接口电路，它使用计数器输出来打开或者关闭停满信号，并且降下或者抬起入口处的起落杆。这个系统的总体框图如图 8.52 所示。

加/减计数器的逻辑框图如图 8.53 所示。它由两个级联加/减十进制计数器组成。下面一段描述它的运算。

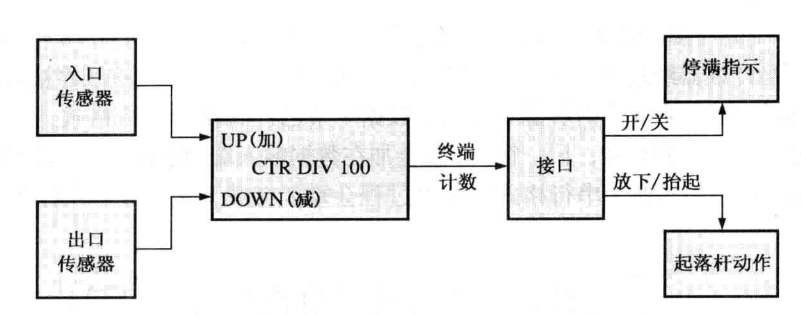


图 8.52 停车库的功能框图

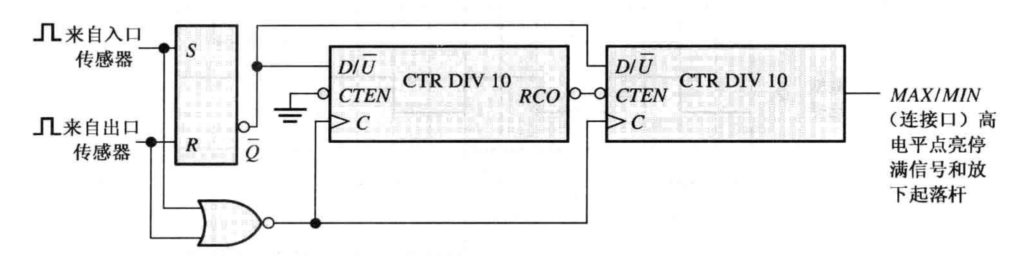


图 8.53 用于停车控制的模 100 加/减计数器的逻辑框图

◇ 计数器递增就是将它的计数值增加 1。

使用并行数据输入把计数器初始预置为 0，这里没有给出。每个进入车库的汽车都会切断一个光束，从而使得传感器产生一个电子脉冲。这个正脉冲的上升沿把 S-R 锁存器置位。锁存器 $\overline{Q}$ 输出上的低电平使得计数器进入加模式。同样，此传感器脉冲经过或非门，并在它后沿的低电平到高电平的转换时计数器加 1。每次汽车进入车库时，计数器就增加 1(递增)。当第 100 辆车进入车库时，计数器就到达它的最后一个状态(100_{10})。*MAX/MIN* 输出变为高电平，并使得接口电路工作(没有细节)，点亮停满信号并降下起落杆以阻止车辆继续进入。

当汽车开出车库时，光电传感器就会产生一个正脉冲，复位 S-R 锁存器并将计数器置于减模式。时钟的后沿将计数减 1(递减)。如果车库停满而车辆离开，那么计数器的 *MAX/MIN* 输出就会变为低电平，并关闭停满信号，同时抬起起落杆。

◇ 计数器递减就是将它的计数值减去 1。

8.7.3 并行数据到串行数据的转换(多路复用)

在第 6 章介绍了使用多路复用和多路分配技术的数据传送的简化例子。本质上，多路复用器输入上的并行数据位被转换为单条传输线上的串行数据位。同时出现在并行线上的一个位组称为并行数据，以时间顺序出现在单线上的位组称为串行数据。

并行到串行的转换通常使用计数器来实现，计数器为数据选择器/多路复用器的数据选择输入提供一个二进制序列，如图 8.54 所示。模 8 计数器的 Q 输出连接到 8 位多路复用器的数据选择输入端。

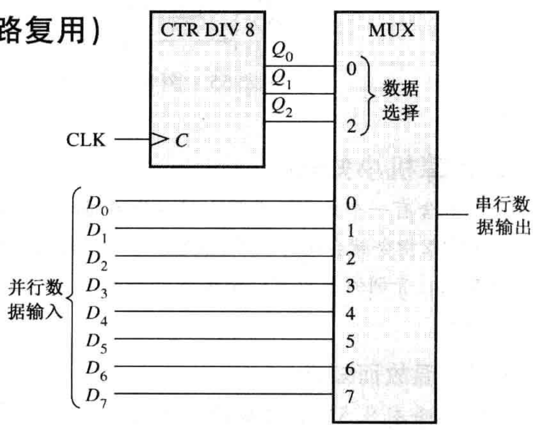


图 8.54 并行数据到串行数据的转换逻辑

图8.55是一个时序图，解释了这个电路的运算。并行数据的第一个字节(8位一组)加到多路复用器的输入。当计数器经过0~7的二进制时序时，开始于D_0的每个位按顺序被选中，并且经过多路复用器到达输出线。在8个时钟脉冲之后，数据字节已经转换为串行格式，并且传送到传输线上。当计数器再循环回到0时，下一个字节就会加在数据输入端，随着计数器循环经过它的8个状态，这个字节又被顺序转换为串行格式。这个过程会重复继续下去，每个并行字节都被转换为串行字节。

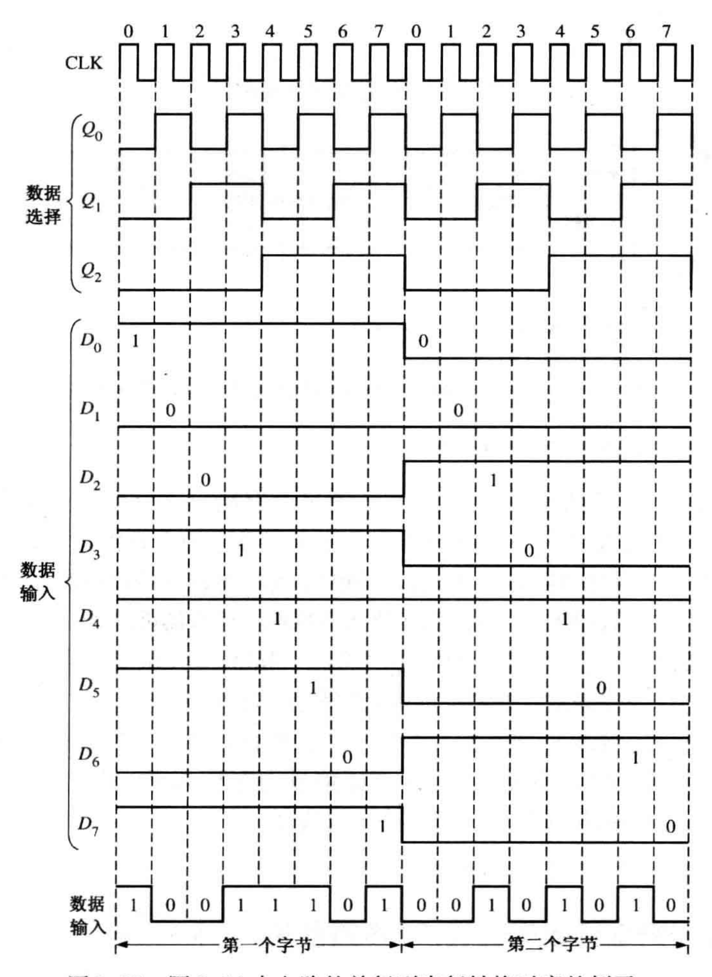


图8.55　图8.54中电路的并行到串行转换时序的例子



计算机小知识

计算机含有一个内部计数器，可以为不同的频率和音调长短编程，由此产生了“音乐”。为了选择特殊的音调，编程指令就会选择一个除数传送到计数器。这个除数设置计数器去除以外部时钟的基率，从而产生声频音调。音调的长短可以由编程指令来设置；因此，使用基本计数器通过控制频率和音调长短来产生美妙的音乐。

8.7节　温故而知新

1. 解释图8.51中与非门的用途。
2. 判断图8.49中小时计数器的两个循环情况，给出使用这两个循环的理由。

8.8 关联标注的逻辑符号

关联标注是ANSI/IEEE标准的基础。关联标注和逻辑符号联合在一起用以指定输入和输出的关系，使得某个给定芯片的逻辑运算可以完全从它的逻辑符号来确定，而不用事先知道其内部结构的细节，也不需要详细的逻辑框图来做参考。介绍具有关联标注的特定逻辑符号，是为了帮助解释将来可能遇到的其他此类符号。

使用74HC163 4位同步二进制计数器来做解释。为了进行比较，图8.56给出了一个传统的块图符号和具有关联标注的ANSI/IEEE符号。此符号和关联标注的基本描述如下。

共用控制块 图8.56(b)中上方有槽口角的块有一些输入和一个输出，它们对于该芯片中所有的组件都被认为是共用的，不是唯一地针对组件中的任何一个。

独立元件 图8.56(b)中下方的块被分成了4个相邻的部分，表示计数器中的4个存储元件(D触发器)，输入为D_0、D_1、D_2和D_3，输出为Q_0、Q_1、Q_2和Q_3。

限定符号 图8.56(b)的标记“CTR DIV 16”把此芯片定义为具有16个状态(DIV 16)的计数器(CTR)。

控制关联(*C*) 如图8.56(b)所示，字母*C*表示控制关联。控制输入常常使能或者禁止存储元件的数据输入(*D*、*J*、*K*、*S*和*R*)。*C*输入常常是时钟输入。在这个例子中，跟随*C*的数字(C5/2，3，4+)表示以前缀5标记的输入依赖于时钟(和时钟同步)。例如，$\overline{CLR}$输入上的5CT=0表示清零功能取决于时钟；也就是说，这是一个同步清零。当$\overline{CLR}$输入为低电平(0)时，计数器就在时钟脉冲的触发边沿复位到零($CT=0$)。同样，标记在存储元件[1]上的5D表示这个数据存储依赖于(同步于)时钟。[1]存储器中的所有标记都适用于下面的[2]、[4]和[8]元件，因而它们是同样类型的标记。

模式关联(*M*) 如图8.56(b)所示，字母*M*表示模式关联。这个标记用来表示不同输入或输出的功能怎样取决于芯片所运行的模式。在这个例子中，这个芯片具有两个运行模式。当$\overline{LOAD}$输入为低电平(0)时，由三角形输入来表示，计数器就处于一个预置的模式(M1)，其中输入数据(D_0、D_1、D_2和D_3)同步载入到4个触发器中。M1中的数字1及标记“1，5 D”中的1给出了一种关联关系，并指示输入数据只在芯片处于预置模式(M1)时才会存储起来，此时$\overline{LOAD}=0$。当$\overline{LOAD}$输入为高电平(1)时，计数器就会向前经过它正常的二进制时序，由M2和“C5/2，3，4+”中的2表示。

与关联(*G*) 如图8.56(b)所示，字母*G*表示与关联，说明由*G*及其后面的一个数字所命名的输入，能够和任何其他标记中具有相同数字前缀的输入及输出执行与运算。在这个特殊的例子中，*ENT*输入上的G3和*RCO*输出上的3CT=15相关，由数字3来指示，并且其关系是与关联，由字母*G*来指示。这就告知如果*RCO*输出是高电平，*ENT*必须是高电平(输入上没有三角形)而计数必须为15($CT=15$)。

同样，标记“C5/2，3，4+”中的数字2、3和4指出当$\overline{LOAD}=1$时，如图所示为模式关联标记M2，计数器顺序经过它的状态，当$ENT=1$和$ENP=1$时，如图所示为与关联标记G3和G4。+指出当这些情况存在时，计数器顺序递增1。

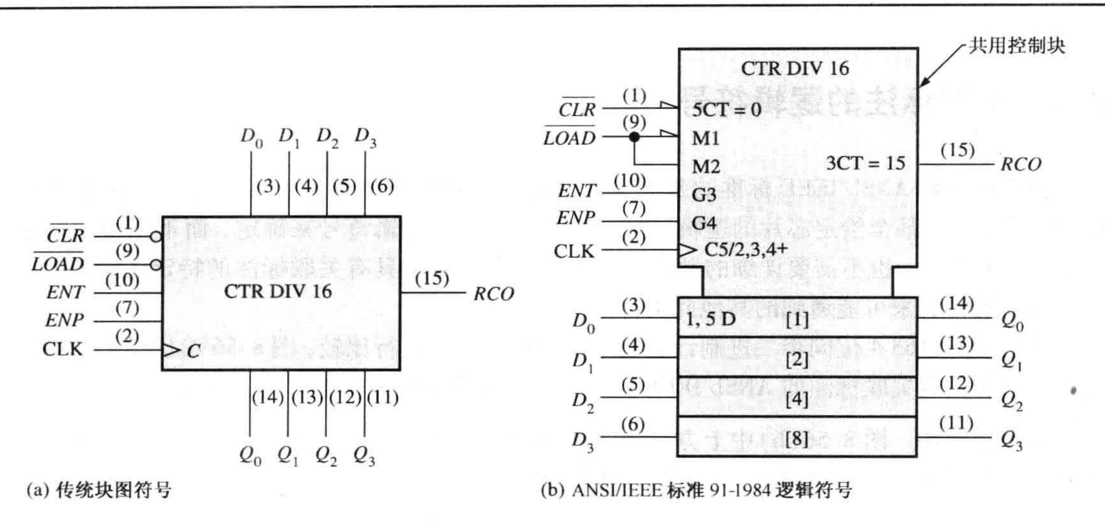


图 8.56 74HC163 4 位同步计数器

8.8 节 温故而知新

1. 在关联标注中，字母 C、M 和 G 表示什么？
2. 哪个字母表示数据存储？



数字系统应用

交通信号灯控制系统的讲解在第 6 章开始，第 7 章继续，系统最后部分的时序逻辑在本章完成，并且所有三个部分组合在一起形成一个完整的系统。图 8.57 给出了系统的框图。

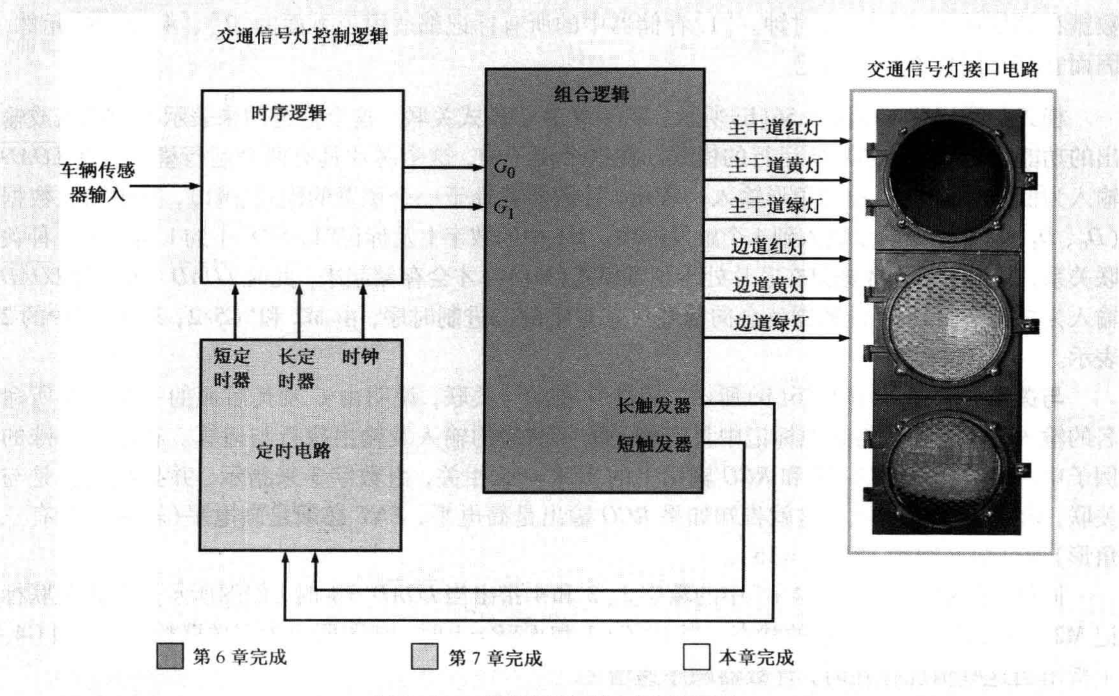


图 8.57 交通信号灯控制系统框图

时序逻辑

时序逻辑电路控制交通信号灯的时序，这些时序基于来自定时电路和边道车辆传感器的输入。时序逻辑电路产生系统的4种状态的2位格雷码时序，这在第6章描述过。

计数器 时序逻辑由一个2位格雷码计数器和相关的输入逻辑组成，如图8.58所示。计数器产生4种状态的输出 G_0 和 G_1 的时序。从一个状态到下一个状态的转换由短定时器(T_S)、长定时器(T_L)和车辆传感器输入(V_s)确定。10 kHz时钟输入(CLK)来自于定时电路中的振荡器。

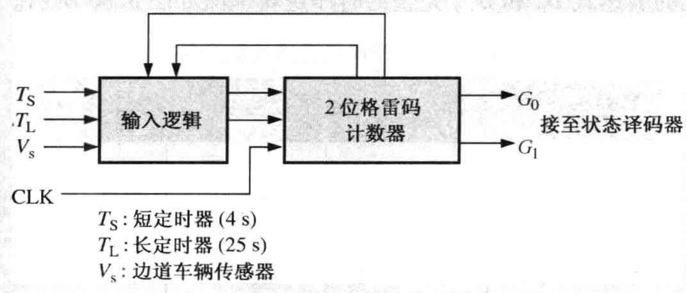


图8.58 时序逻辑框图

图8.59给出了可用来实现格雷码计数器的两个D触发器。计数器控制逻辑的输出提供D触发器的输入，这样就得到了合适的时序。

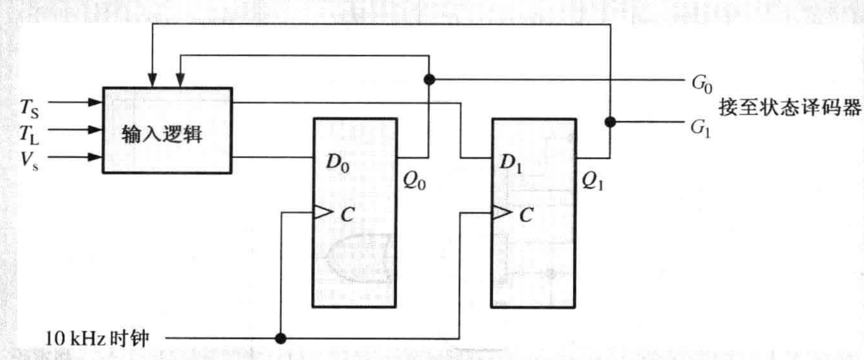


图8.59 时序逻辑框图，使用两个D触发器实现的计数器

D触发器的转换表如表8.14所示。从第6章的状态图得到如表8.15所示的次态表。

表8.14 D触发器转换表

输出转换			触发器输入
Q_N		Q_{N+1}	D
0	⟶	0	0
0	⟶	1	1
1	⟶	0	0
1	⟶	1	1

计数器控制逻辑 从表8.14和表8.15中可以确定每个触发器进入1状态所需要的逻辑条件。例如，当前状态是00时，G_0从0变化到1。输入条件在表8.15的第二行给出。D_0必须为1以使 G_0 变为1，或在下一个时钟脉冲到来时保持为1。对于 D_0为1的情况，可以根据表8.15写出布尔表达式：

$$D_0 = \bar{G}_1\bar{G}_0\bar{T}_LV_s + \bar{G}_1G_0T_s + \bar{G}_1G_0\bar{T}_s + G_1G_0T_LV_s$$

去掉表达式中间两项中的变量 T_s 和 $\bar{T}_s$，得到表达式：

$$D_0 = \overline{G}_1\overline{G}_0\overline{T}_LV_s + \overline{G}_1G_0 + G_1G_0T_LV_s$$

1. 陈述在 D_0 的表达式中允许取消 T_s 的布尔定理和规则。

2. 使用卡诺图进一步简化 D_0 的表达式以得到最小形式。同样，可以从表 8.15 得到 D_1 的表达式：

$$D_1 = \overline{G}_1G_0\overline{T}_s + G_1G_0T_LV_s + G_1G_0\overline{T}_L + G_1G_0\overline{V}_s + G_1\overline{G}_0T_s$$

3. 使用布尔定理、规则和/或者卡诺图简化 D_1 的表达式以得到最小形式。

4. 基于最小化的表达式 D_0 和 D_1，完整的时序逻辑框图如图 8.60 所示。

表 8.15 时序逻辑转换的次态表

当前状态		次态			FF 触发器	
Q_1	Q_0	Q_1	Q_0	输入条件	D_1	D_0
0	0	0	0	$T_L+\overline{V}_s$	0	0
0	0	0	1	$\overline{T}_LV_s$	0	1
0	1	0	1	T_S	0	1
0	1	1	1	$\overline{T}_S$	1	1
1	1	1	1	$T_L\ V_s$	1	1
1	1	1	0	$\overline{T}_L+\overline{V}_s$	1	0
1	0	1	0	T_S	1	0
1	0	0	0	$\overline{T}_S$	0	0

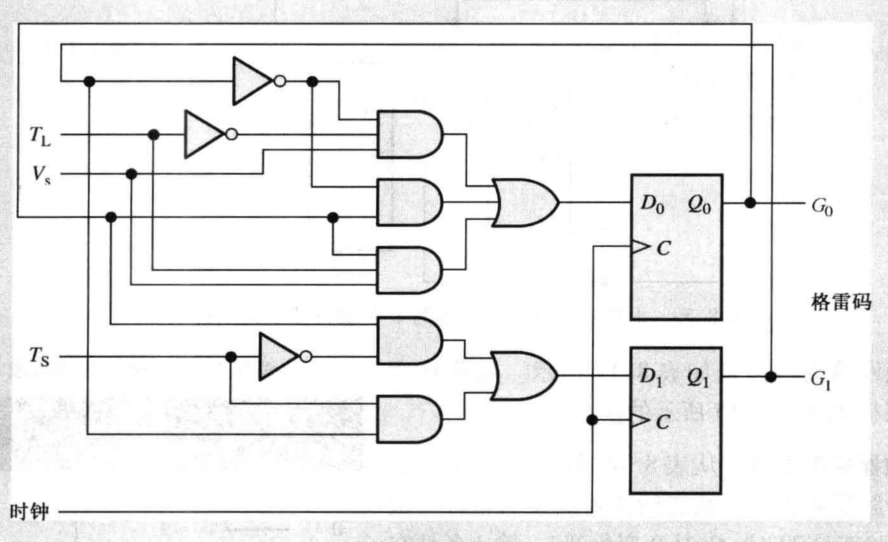


图 8.60 时序逻辑框图

计算机仿真

图 8.61 给出了交通信号灯控制系统的时序逻辑部分的计算机仿真界面。

打开在线资源系统应用文件夹中的文件 SAA08A。使用 Multisim 软件运行时序逻辑仿真程序。所有用于仿真目的输入以开关输入的形式给出。设置 T_L、T_S 和 V_s 的输入条件。然后，以很短的时间使时钟输入为低电平(地)，再回到高电平(V_{CC})。跟随状态图给出的时序。

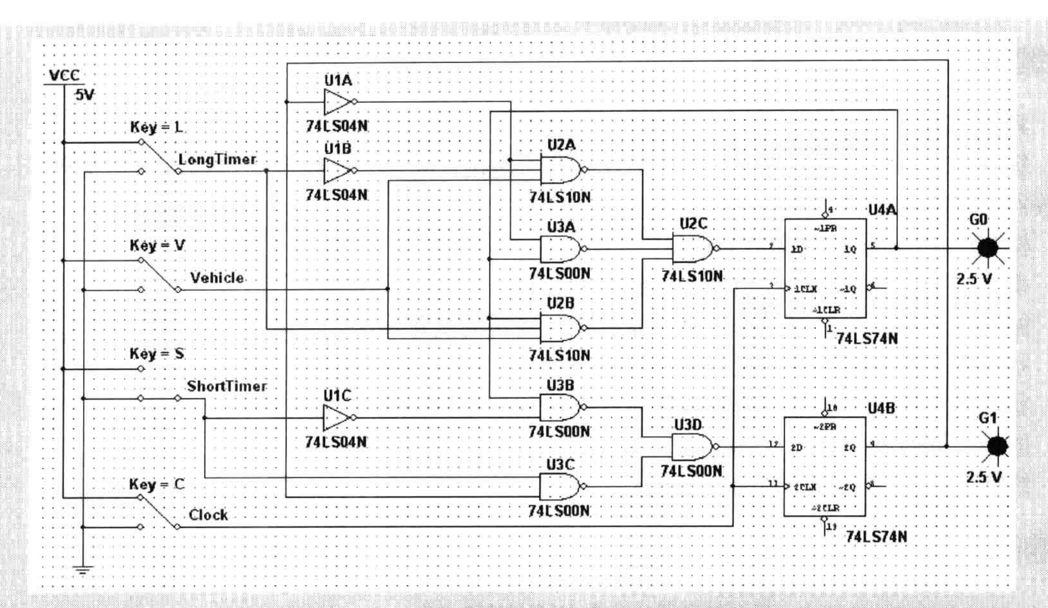


图 8.61　时序逻辑计算机仿真的 Multisim 界面。开关用于测试目的。当探针是红色时，指示 1

完整的交通信号灯控制系统

现在系统的所有三个部分都已设计完成，合起来形成一个完整的系统，由图 8.57 的框图给出。图 8.62 给出了整个系统的计算机仿真界面。系统的每一个部分转换成一个 Multisim 子电路，以便降低屏幕显示的复杂性。

主干道的绿灯将一直点亮，直到边道没有车辆为止。当边道有车辆出现时，主干道的绿灯熄灭，并且转为红灯，持续时间 25 秒，以便让边道的车辆通行。如果边道持续有车辆，那么红灯和绿灯每隔 25 秒转换一次。

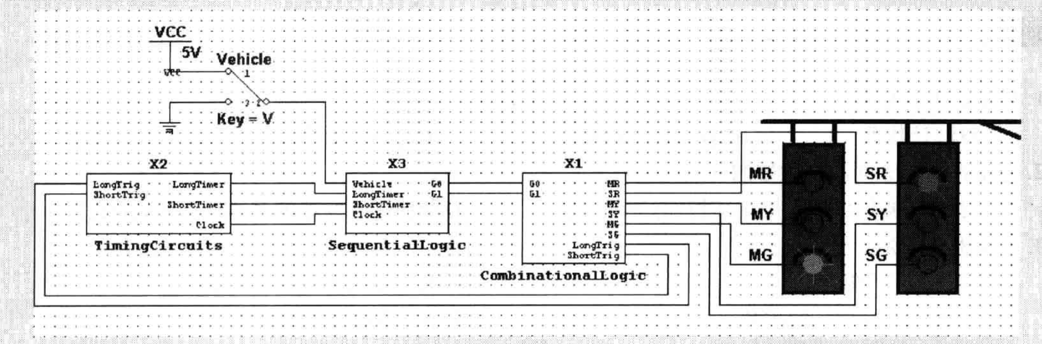


图 8.62　每个部分的子电路合在一起的交通信号灯控制系统的计算机仿真

打开系统应用示例中的文件 SAA08B。使用 Multisim 软件运行交通信号灯控制系统程序，观察它的运行。第一次打开时信号灯随机点亮。仿真时间有差别。

把知识用于实践

添加第 6 章设计的人行道输入，运行仿真程序。

关键词

异步 不在同一时间发生。

级联 端口与端口的连接，如几个计数器的连接，从一个计数器的终端输出连接到下一个计数器的使能输入端。

模 10 10 种状态或变量的特性。

模 一个计数器序列经历的不同状态的数目。

循环 从最后或最终状态回到初始状态所经历的变化。

状态图 描述状态或变量序列的图。

状态机 通过外部和内部输入展示状态序列的逻辑系统。

同步 同时发生。

终端 一个计数器序列的最后状态。

判断题 (答案在本章的结尾。)

1. 在一个异步计数器中，所有触发器的状态在同一时间变化。
2. 在一个同步计数器中，时钟同时作用在所有的触发器上。
3. 一个同步计数器也可以看做行波计数器。
4. 一个十进制计数器有 16 种状态。
5. 一个具有四级的计数器，最大的模为 16。
6. 为了获得最大模 32，需要 16 级。
7. 如果当前状态为 1000，那么一个在减模式时的 4 位加/减计数器的下一个状态是 0111。
8. 两个十进制计数器级联，可以把时钟频率除以 20。
9. 一个具有截断序列的计数器，其计数小于它的最大状态数。
10. 为了获得 100 的模，需要 10 个十进制计数器。

自测题 (答案在本章的结尾。)

1. 异步计数器称为
   (a)行波计数器 (b)多时钟计数器 (c)十进制计数器 (d)模计数器
2. 异步计数器和同步计数器的区别是
   (a)时序中状态的个数 (b)时钟脉冲的方法 (c)所使用触发器的类别 (d)模的数值
3. 计数器的模是
   (a)触发器的个数 (b)时序中实际的状态个数
   (c)一秒钟内再循环的次数 (d)状态的最大可能个数
4. 3 位二进制计数器的最大模是
   (a)3 (b)6 (c)8 (d)16
5. 4 位二进制计数器的最大模是
   (a)16 (b)32 (c)8 (d)4
6. 模 12 计数器必须具有
   (a)12 个触发器 (b)3 个触发器 (c)4 个触发器 (d)同步时钟
7. 下面的哪一个计数器具有截断模?
   (a)模 8 (b)模 14 (c)模 16 (d)模 32
8. 一个由触发器组成的 4 位异步(行波)计数器，每个计数器从时钟到 Q 输出的时间延迟为 12 ns。计数

器从 1111 再循环回到 0000，它花费的总时间为

(a) 12 ns　(b) 24 ns　(c) 48 ns　(d) 36 ns

9. BCD 计数器是哪种计数器的例子？

(a) 全模计数器　(b) 十进制计数器　(c) 截断模计数器　(d) 答案(b)和(c)

10. 下面的哪一个状态在 8421 BCD 计数器中是无效状态？

(a) 1100　(b) 0010　(c) 0101　(d) 1000

11. 三个级联的模 10 计数器的整体模是

(a) 30　(b) 100　(c) 1000　(d) 10 000

12. 一个 10 MHz 的时钟频率应用在一个级联计数器上，该级联计数器有一个模 5 计数器、模 8 计数器和两个模 10 计数器。最低输出频率可能是

(a) 10 kHz　(b) 2.5 kHz　(c) 5 kHz　(d) 25 kHz

13. 一个 4 位二进制加/减计数器处于二进制状态 0。那么在减模式中的下一个状态是

(a) 0001　(b) 1111　(c) 1000　(d) 1110

14. 模 13 二进制计数器的终端计数是

(a) 0000　(b) 1111　(c) 1101　(d) 1100

习题

8.1 节　异步计数器运算

1. 对于如图 8.63 所示的异步计数器，对于 8 个时钟脉冲给出完整的时序图，同时给出时钟 Q_0 和 Q_1 的波形。

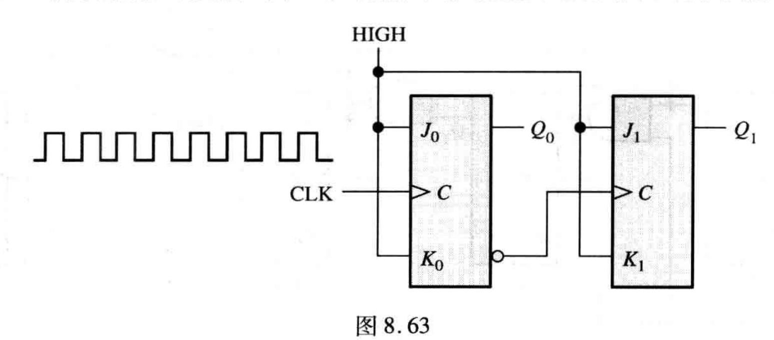


图 8.63

2. 对于图 8.64 中的异步计数器，对于 6 个时钟脉冲给出完整的时序图。给出时钟 Q_0、Q_1 和 Q_2 的波形。

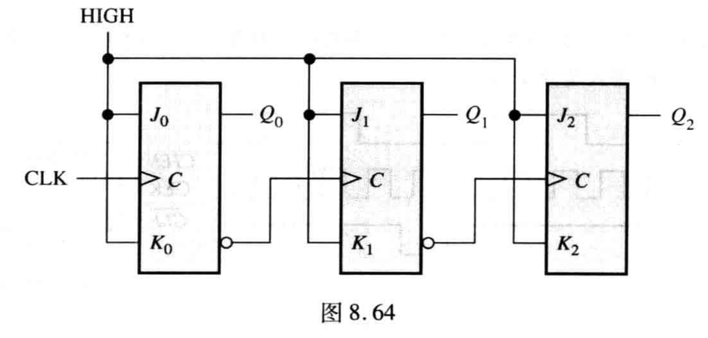


图 8.64

3. 在习题 2 的计数器中，假设每个触发器从时钟的触发边沿到 Q 输出的变化都有一个 8 ns 的传输延迟。在给定的状态下，确定从时钟脉冲到计数器之间的最差状态(最长)延迟时间。指定状态或最差延迟发生时的状态。

4. 给出怎样为下面的每一个模连接 74LS93 4 位异步计数器：

(a)9　(b)11　(c)13　(d)14　(e)15

8.2 节　同步计数器运算

5. 如果习题 3 中的计数器为同步而不是异步的，那么最长延迟时间应当是多少?

6. 为图 8.65 中的 5 级同步二进制计数器给出完整的时序图。验证 Q 输出的波形表示每个时钟脉冲后的正确二进制数。

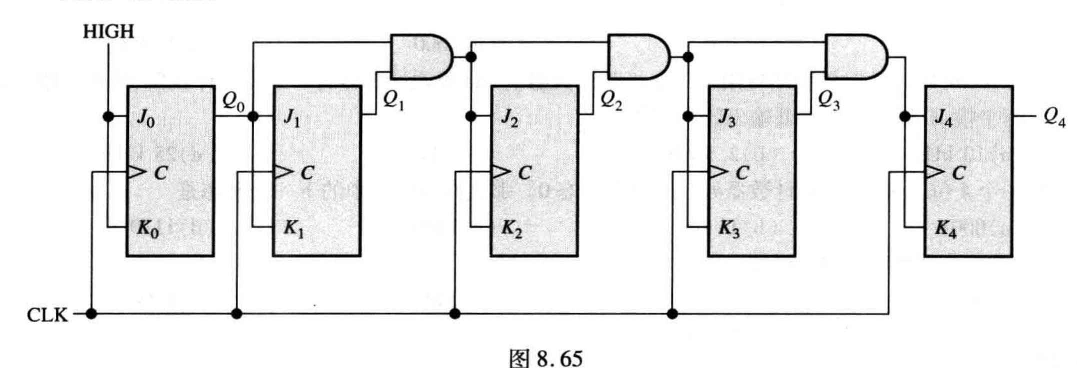


图 8.65

7. 通过分析每个时钟脉冲之前的每个触发器的 J 和 K 输入，证明图 8.66 中的十进制计数器经过一个 BCD 序列。解释每种情况下的这些条件怎样使计数器进入下一个正确的状态。

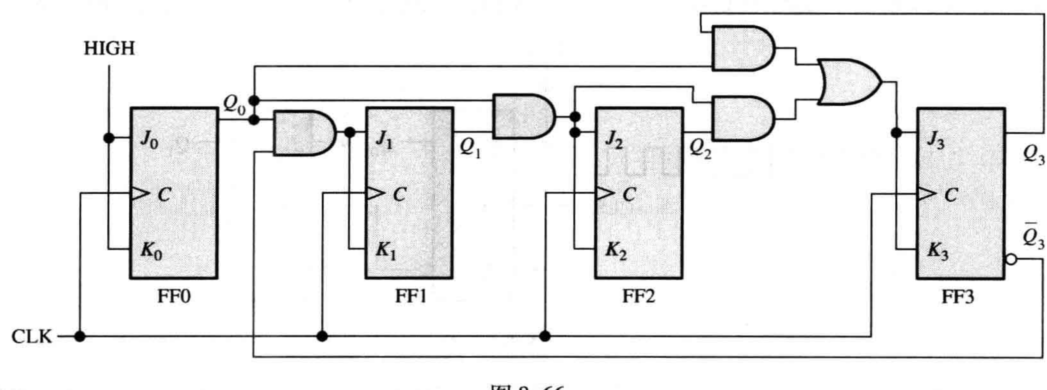


图 8.66

8. 图 8.67 中的波形应用于计数使能端、清零及时钟输入，如图所示。给出正确的和这些输入相关的计数器输出波形。清零输入是异步的。

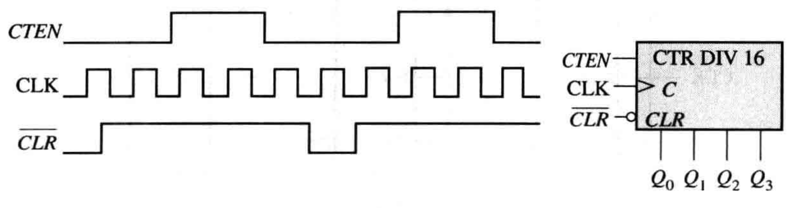


图 8.67

9. BCD 十进制计数器如图 8.68 所示。波形按照所指示的那样应用于时钟和清零输入。确定每个计数器输出(Q_0、Q_1、Q_2和 Q_3)的波形。清零是同步的，并且计数器初始时处于二进制 1000 状态。

10. 图 8.69 中的波形应用于一个 74HC163 计数器中，确定 Q 输出和 RCO。输入为 $D_0=1$、$D_1=1$、$D_2=0$ 和 $D_3=1$。

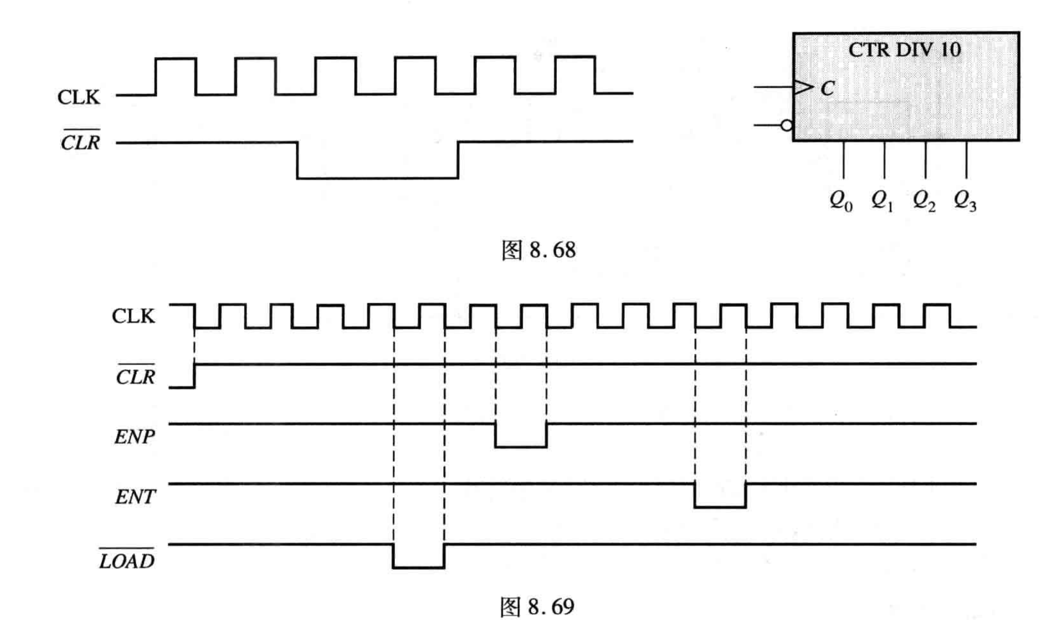


图 8.68

图 8.69

11. 图 8.69 中的波形应用于一个 74HC161 计数器中。确定 Q 输出和 RCO。输入为 $D_0=1$、$D_1=0$、$D_2=0$ 和 $D_3=1$。

8.3 节　加/减同步计数器

12. 为经过下面时序的 3 位加/减计数器给出完整的时序图。说明计数器什么时候处于加模式，什么时候处于减模式。假设为上升沿触发。

0, 1, 2, 3, 2, 1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1, 0

13. 为具有如图 8.70 所示输入波形的 74HC190 加/减计数器，绘制出 Q 输出波形。一个二进制 0 位于数据输入上。开始于计数 0000。

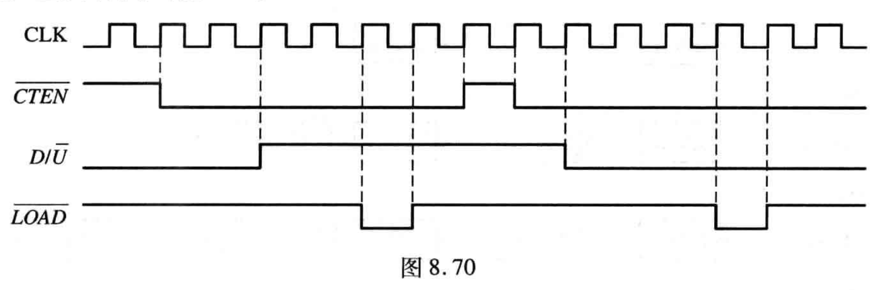


图 8.70

14. 如果加/减（$D/\overline{U}$）输入信号反相，其他输入不变，重做习题 13。

15. 如果 $\overline{CTEN}$ 反相，其他输入不变，重做习题 13。

8.4 节　同步计数器的设计

16. 确定图 8.71 中计数器的时序。

17. 确定图 8.72 中计数器的时序。计数器开始时为清零状态。

18. 设计一个计数器以产生下面的时序。使用 J-K 触发器。

00, 10, 01, 11, 00, …

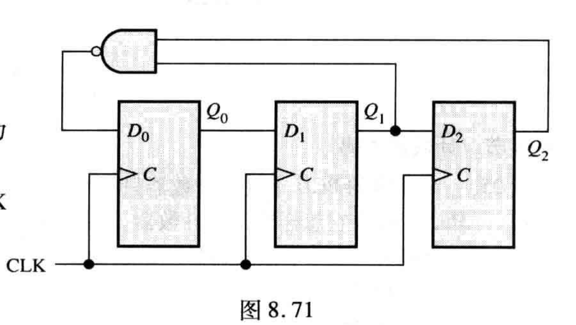


图 8.71

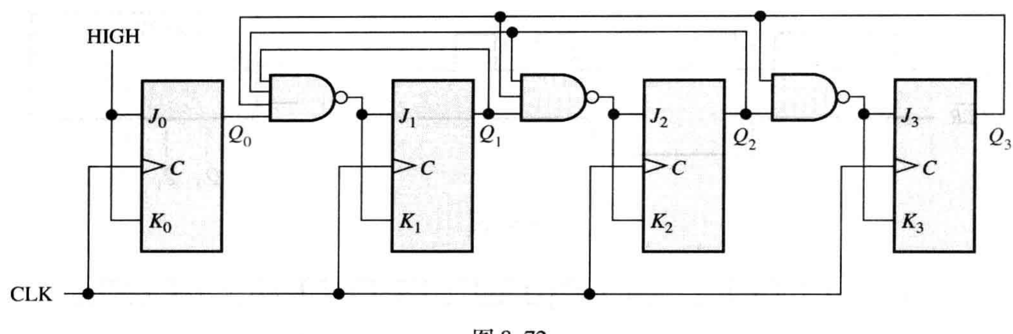


图 8.72

19. 设计一个计数器以产生下面的时序。使用 J-K 触发器。

1, 4, 3, 5, 7, 6, 2, 1, …

20. 设计一个计数器以产生下面的时序。使用 J-K 触发器。

0, 9, 1, 8, 2, 7, 3, 6, 4, 5, 0, …

21. 设计一个计数器，具有图 8.73 中状态图所给出的时序。

加

减

0, 3, 5, 7, 9, 11

图 8.73

8.5 节 级联计数器

22. 对于图 8.74 中的级联计数器配置，确定由圆圈数字所表示的每个波形的频率，并且确定整体的模。

23. 扩展图 8.39 中的计数器以创建一个 10 000 分频的计数器和一个 100 000 的计数器。

24. 利用一般框图，给出怎样从一个 10 MHz 的时钟中得到下面的频率，使用单个触发器、模 5 计数器和十进制计数器：

(a) 5 MHz (b) 2.5 MHz (c) 2 MHz (d) 1 MHz (e) 500 kHz
(f) 250 kHz (g) 62.5 kHz (h) 40 kHz (i) 10 kHz (j) 1 kHz

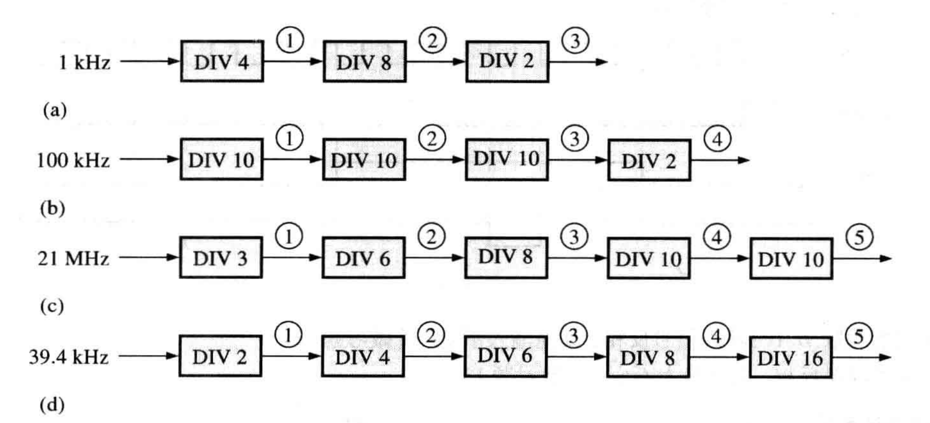


图 8.74

8.6 节 计数器译码

25. 一个给定的 BCD 十进制计数器只有 Q 输出可用，给出对于下面的每一个状态译码，需要什么译码逻辑，并给出它怎样连接到计数器。每个译码需要一个高电平输出指示。最高有效位在左边。

(a)0001 (b)0011 (c)0101 (d)0111 (e)1000

26. 对于连接到图 8.75 中译码器的 4 位二进制计数器，确定和时钟脉冲相关的每个译码器的输出波形。

27. 如果图 8.75 中的计数器是异步的，确定在译码器输出波形上的什么地方发生译码假信号。

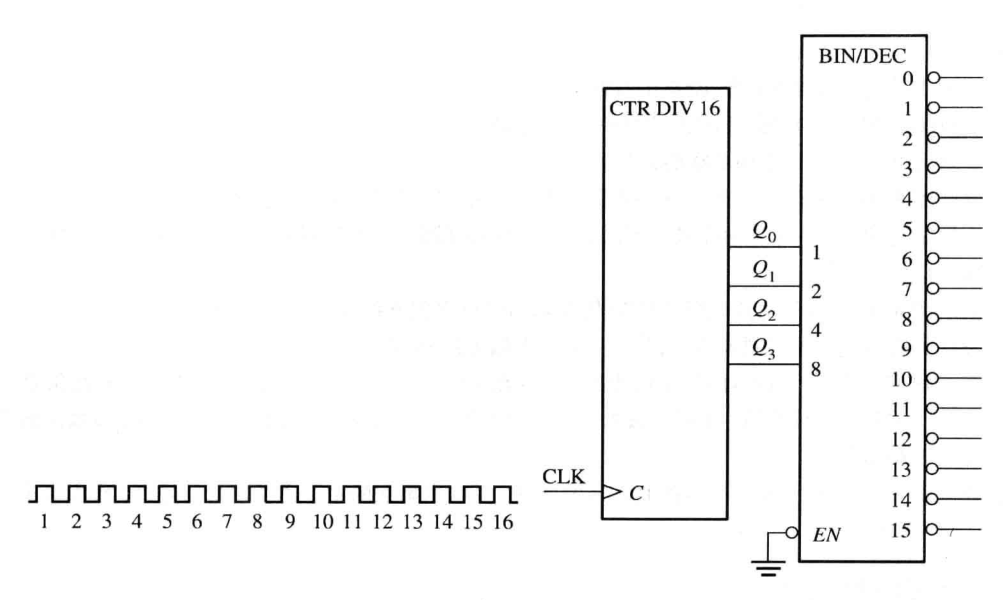


图 8.75

28. 修改图 8.76 中的电路，以消除译码假信号。
29. 分析图 8.43 中的计数器，观察在译码门输出上是否有假信号。如果发生假信号，建议一种方式来消除它们。
30. 分析图 8.44 中的计数器，看看在译码门输出上是否有假信号。如果发生假信号，改变设计以消除它们。

8.7 节　计数器应用

31. 假设图 8.49 中的数字时钟初始时被复位到了 12 点。确定在发生了 62 个 60 Hz 的脉冲之后，每个计数器的二进制状态。
32. 在图 8.49 中的数字时钟电路中，每个计数器的输出频率是多少？
33. 对于图 8.52 所示的停车控制系统，在一个给定的 24 小时期间，入口和出口传感器脉冲的图样如图 8.76 所示。如果在该时期之前车库中已经有 53 辆车，那么在 24 小时结束之后计数器的状态是什么？

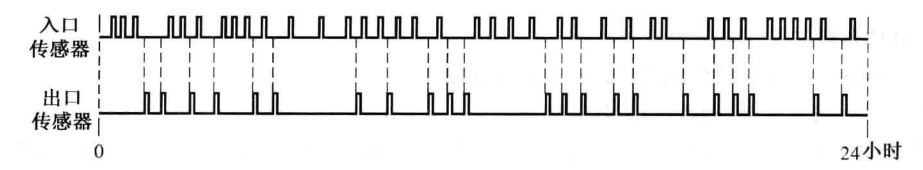


图 8.76

34. 图 8.54 中对应十进制数的二进制数（D_0 为最低有效位）出现在并行-串行转换器的并行数据输入。计数器的初始值全部为零，使用 10 kHz 的时钟。画出时序图，给出时钟、计数器输出和串行数据输入。

数字系统应用

35. 仅使用与非门设计出交通信号灯控制系统的时序部分的输入逻辑。
36. 用 D 触发器替代图 8.59 的 J-K 触发器组成的 2 位格雷码状态计数器。
37. 给出如何把 25 秒绿灯亮的时间间隔改为 60 秒的方法。

特殊设计问题

38. 使用十进制计数器设计模 1000 的计数器。
39. 修改图 8.42 的计数器，得到模 30 000 的计数器。
40. 重复习题 39，得到模 50 000 的计数器。
41. 修改图 8.49、图 8.50、图 8.51 中的数字时钟，使得它可以复位到任何时间。
42. 为数字时钟设计一个闹钟电路，可以检测一个预先设定的时间(只需要小时和分钟)，产生一个声音报警信号。
43. 修改图 8.53 的设计电路，用于 1000 辆车辆的停车库和 3000 辆车辆的停车库。
44. 设计具有固定功能的图 8.54 的并行 - 串行数据传送逻辑电路。
45. 在习题 17 中发现计数器出现锁死情况，并且在两个状态之间变化。说明这样的运行结果是设计的错误。重新设计计数器，使得计数器进入第二个循环状态，使得计数器在下一个时钟脉冲到来时循环进入全零状态。
46. 修改图 8.57 中的交通信号灯控制系统的框图，反映出附加的主干道左转弯信号灯，在人行道绿灯亮时亮 15 秒。

WWW Multisim 故障诊断练习

47. 打开文件 P08-57，测试 4 位异步计数器以确定是否有问题，如果有，尽可能地判断出来。
48. 打开文件 P08-58，测试 3 位同步计数器以确定是否有问题，如果有，尽可能地判断出来。
49. 打开文件 P08-59，测试 BCD 计数器以确定是否有问题，如果有，尽可能地判断出来。
50. 打开文件 P08-60，测试 74163 4 位二进制计数器以确定是否有问题，如果有，尽可能地判断出来。
51. 打开文件 P08-61，测试 74190 加/减十进制计数器以确定是否有问题，如果有，尽可能地判断出来。

答案

温故而知新

8.1 节　异步计数器运算

1. 异步的意思是第一个触发器工作以后，每个触发器由它的前一个触发器使能。
2. 一个模 14 计数器有 14 个状态，需要 4 个触发器。

8.2 节　同步计数器运算

1. 在一个同步计数器中时钟是同时作用于触发器的。
2. 计数器可以被预置位(初始化)到任意状态。
3. 当 ENP 和 ENT 都是高电平时，计数器使能；当最后一个状态顺序到达时，RCO 变为高电平。

8.3 节　加/减同步计数器

1. 计数器进入 1001。
2. 加：1111；减：0000；下一个状态是 1111。

8.4 节　同步计数器的设计

1. $J=1$，$K=X$(“无关”)。
2. $J=X$(“无关”)，$K=0$。
3. (a)下一个状态是 1011。

   (b)Q_3(MSB)：无变化或置位；Q_2：无变化或复位；Q_1：无变化或置位；
   Q_0(LSB)：置位或切换。

8.5 节 级联计数器

1. 3 个十进制计数器产生 ÷1000；4 个十进制计数器产生 ÷10 000。
2. (a) ÷20：触发器和 DIV 10；
   (b) ÷32：触发器和 DIV 16；
   (c) ÷160：DIV 16 和 DIV 10；
   (d) ÷320：DIV 16、DIV 10 和触发器。

8.6 节 计数器译码

1. (a)因为没有信号位改变，所以没有状态变化。
   (b)0000，0001，0010，0101，0110，0111
   (c)因为没有信号位改变，所以没有状态变化。
   (d)0001，0010，0011，0100，0101，0110，0111，1000，1001，1010，1011，1100，1101，1110

8.7 节 计数器应用

1. 计数到 9 以后的第一个时钟使得门 G_1 复位。门 G_2 译码计数 12，预置计数器到 0001。
2. 小时十进制计数器经过 0 ~ 9 的每个状态，当它从 9 循环回到 0 时，触发器切换到置位状态。这样就显示 10。当小时十进制计数器处于状态 12 时，与非门译码使得计数器在下一个时钟脉冲时循环到状态 1。触发器复位。显示器显示结果数 1(01)。

8.8 节 关联标注的逻辑符号

1. C：控制，通常指时钟；M：模；G：与。
2. D 表示数据存储。

例题的相关问题

8.1 参见图 8.77。

8.2 连接 Q_0 到与非门作为第三个输入（Q_2 和 Q_3 是其他两个输入）。连接 $\overline{CLR}$ 线到 FF0、FF2 和 FF3 的 $\overline{CLR}$ 输入。

8.3 参见图 8.78。

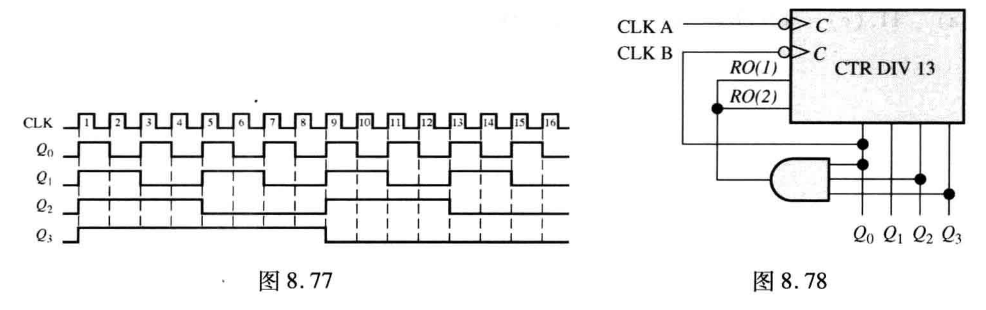


图 8.77 图 8.78

8.4 参见图 8.79。

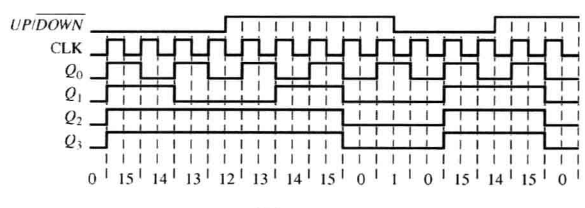


图 8.79

8.5 参见表8.16。

8.6 对图8.35的逻辑应用布尔代数，给出每个或门的输出和步骤5的表达式一致。

表8.16

当前状态			J、K输入						NEXT状态			
Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0	Q_2	Q_1	Q_0	
0	0	0	0	0	1	1	1	1	0	1	1	
0	1	1	1	1	1	1	1	1	1	0	0	
1	0	0	0	0	1	1	1	0	1	1	1	无效状态
1	1	0	1	1	1	1	1	0	0	0	1	无效状态

8.7 $10^5 = 100\ 000$，需要5个十进制计数器。

8.8 $f_{Q_0} = 1\ \text{MHz}/[(10)(2)] = 50\ \text{kHz}$。

8.9 参见图8.80。

8.10 置数为$8AC0_{16}$。$16^4 - 8AC0_{16} = 65\ 536 - 32\ 520 = 30\ 016$。

$f_{TC4} = 10\ \text{MHz}/30\ 016 = 333.2\ \text{Hz}$。

8.11 参见图8.81。

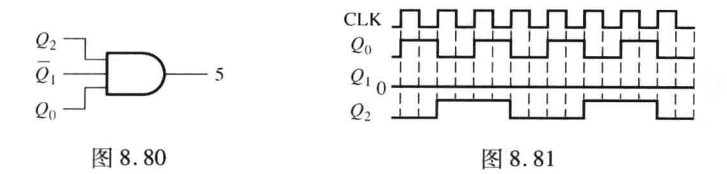


图8.80 图8.81

判断题

1. F 2. T 3. T 4. F 5. T 6. F 7. T 8. F 9. T 10. F

自测题

1. (a) 2. (b) 3. (b) 4. (c) 5. (a) 6. (c) 7. (b) 8. (c) 9. (d)
10. (a) 11. (c) 12. (b) 13. (b) 14. (d)

第9章　移位寄存器

章节提纲

9.1　基本移位寄存器的功能

◇ 寄存器可以由一个或者多个用以存储和移位数据的触发器组成。

寄存器是一个具有两种基本功能的数字电路：数据存储和数据移动。寄存器的存储能力使得它成为一种重要的存储器元件。图9.1解释了在D触发器中存储1或者0这个存储概念。一个1加在数据输入上，如图所示，同时加一个时钟脉冲使触发器置位，这样就存储这个1。当输入上的1被移走之后，触发器还是保持在置位状态，因此存储了1。应用相似的过程，通过复位触发器来存储0，如图9.1所示。

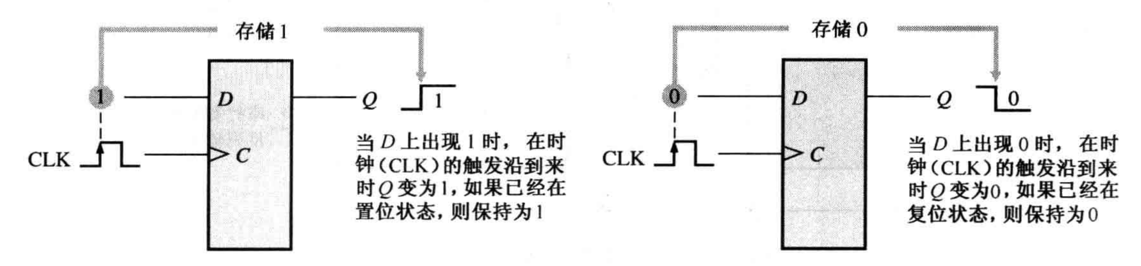


图9.1　触发器作为存储元件

寄存器的存储容量是它可以包含的数字数据的总位数(1和0)。移位寄存器中的每一级(触发器)都表示存储容量中的一位；所以，寄存器中的级数决定了它的存储容量。

寄存器的移位能力允许在寄存器内数据从级到级的移动，或者根据所加的时钟脉冲，数据进入或者离开寄存器。图9.2阐释了移位寄存器中的数据类型。方块图表示任意的4位寄存器，箭头指示数据移动的方向。

9.1节　温故而知新　(答案在本章的结尾。)

1. 通常情况下，计数器和移位寄存器的区别是什么？
2. 移位寄存器的两个主要功能是什么？

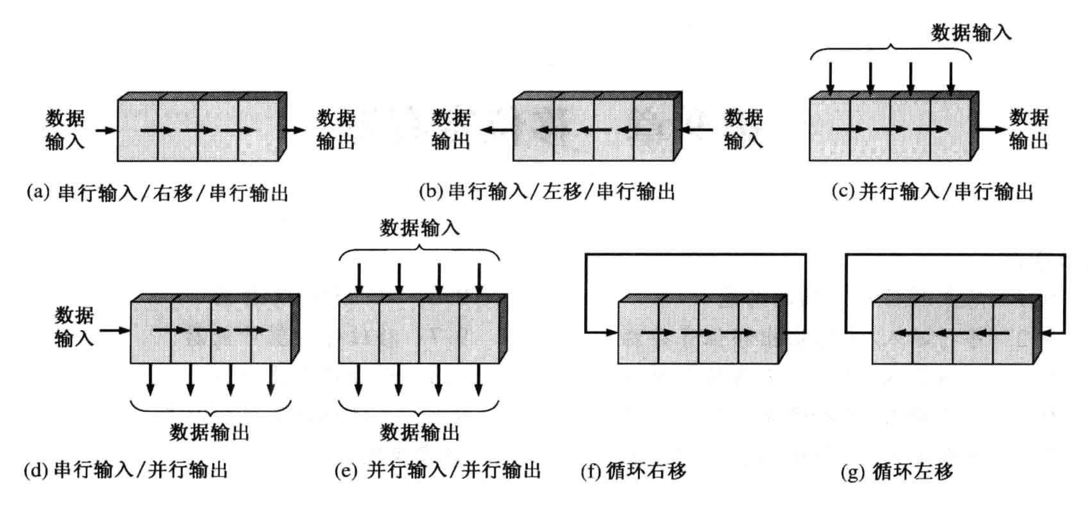


图 9.2 移位寄存器的基本数据移动(用4位做解释。位按照箭头所指的方向移动)

9.2 串行输入/串行输出移位寄存器

首先来看串行进入一个典型的移位寄存器的数据。图 9.3 给出了一个用 D 触发器实现的 4 位寄存器。其中有 4 个级，这个寄存器可以存储 4 位数据。

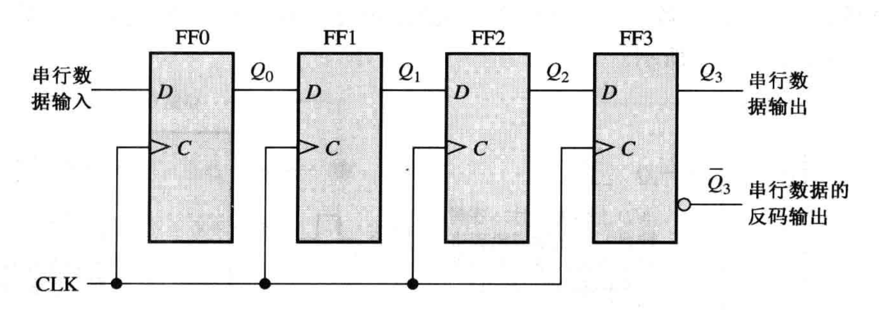


图 9.3 串行输入/串行输出移位寄存器



计算机小知识

经常需要清零计算机中的内部寄存器。例如，可能先要在算术运算或者其他运算之前使寄存器清零。计算机中寄存器清零的一种方法是，使用软件减去寄存器自身所包含的内容。当然结果总是为 0。例如，执行这个运算的计算机指令是 SUB AL, AL。使用这个指令，名称为 AL 的寄存器就被清零了。

图 9.4 解释了 4 位的 1010 进入寄存器的情况，开始于最右边的位。寄存器初始时为清零。0 首先被置入数据输入线上，使得 FF0 的 $D=0$。当第一个脉冲到来时，FF0 被复位，因此存储 0。

接下来是第二位，也就是 1，被加到数据输入上，使得 FF0 的 $D=1$，而 FF1 的 $D=1$，这是因为 FF1 的 D 输入连接 Q_0 输出。当第二个时钟脉冲到来时，数据输入上的 1 被移位到 FF0，使得 FF0 被置位；并且 FF0 的 0 被移位到 FF1。

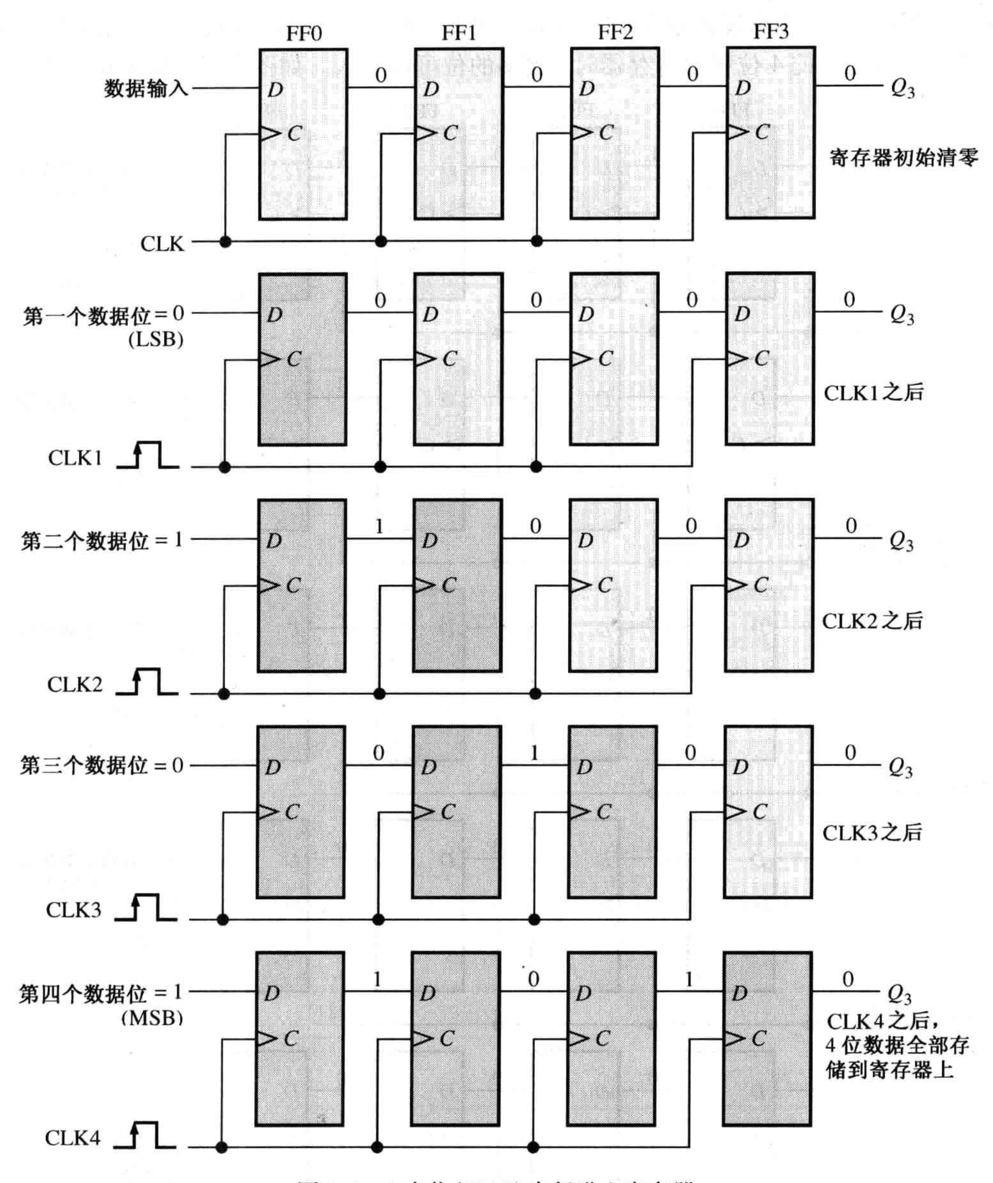


图 9.4　4 个位(1010)串行进入寄存器

第三位，也就是 0，现在被置入数据输入线上，同时加了一个时钟脉冲。这个 0 进入 FF0，FF0 中存储的 1 移位到 FF1 中，而 FF1 所存储的 0 移位到 FF2 中。

最后一位，也就是 1，现在被加到数据输入上，同时加了一个时钟脉冲。这次 1 进 FF0，而存储在 FF0 中的 0 移位到 FF1 中，存储在 FF1 中的 1 移位到 FF2 中，存储在 FF2 中的 0 移位到 FF3 中。这就完成了 4 位串行进入移位寄存器的过程，在寄存器中它们可以被存储无限长的时间，只要触发器有直流电源连接。

◇ 对于串行数据来说，一次传送一位。

如果想要得到寄存器输出的数据，数位就必须串行移出并在 Q_3 输出取走，如图 9.5 所示。在刚描述的数据输入运算中的 CLK4 之后，最右边的位(LSB)0 就会出现在 Q_3 输出。当时钟脉

冲 CLK5 到来时，第二位出现在 Q_3输出。时钟脉冲 CLK6 把第三位移到输出，而 CLK7 把第四位移位到输出。当原始的 4 位移出寄存器时，更多的位可以移入。如图所示，全部移入 0。

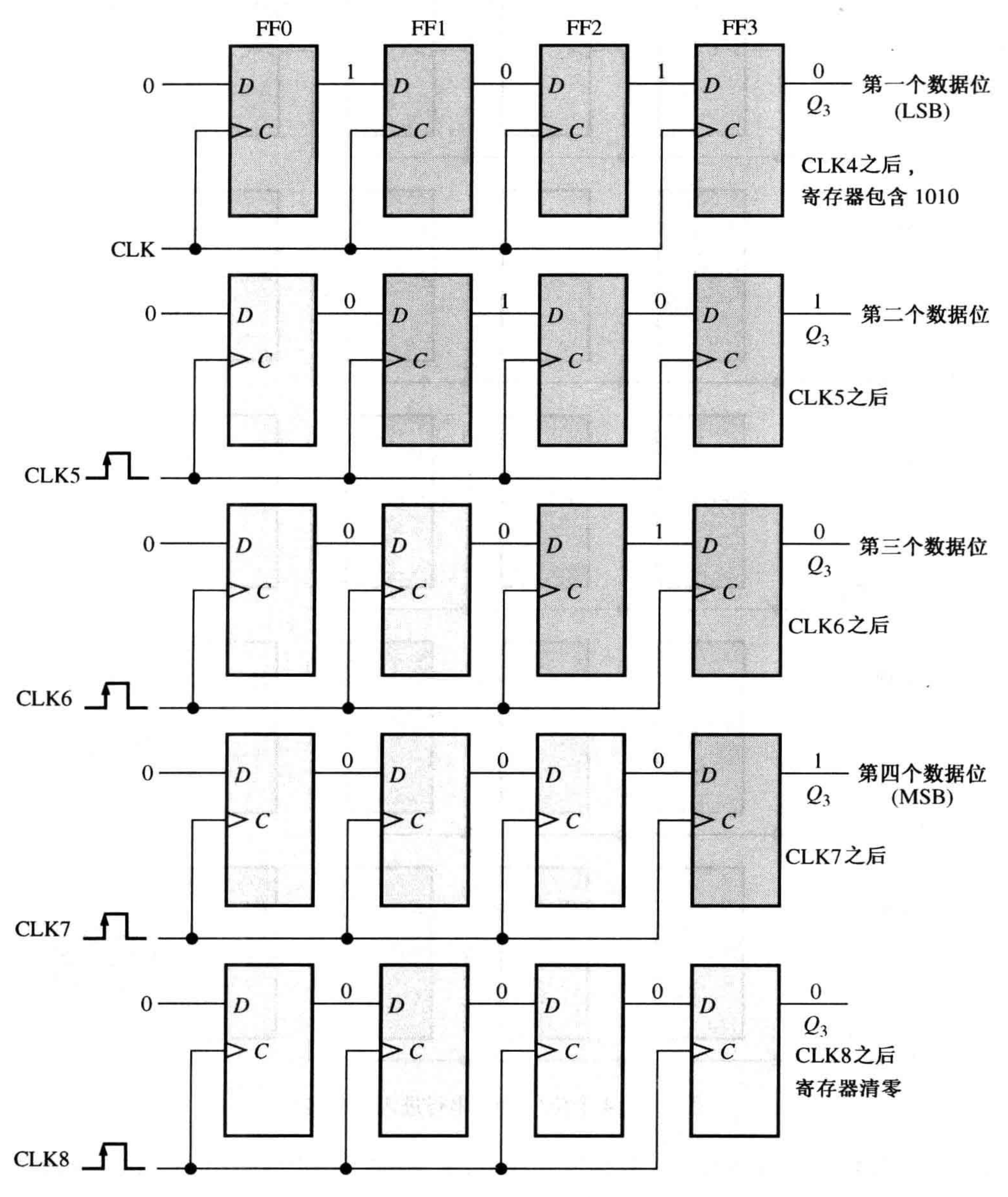


图 9.5　4 位(1010)串行输出寄存器，然后全部用 0 替代

例 9.1　对于指定的数据输入和时钟波形，给出图 9.6(a) 中的 5 位寄存器的状态。假设该寄存器初始时被清零(全 0)。

解： 第一个数据位(1)在第一个时钟脉冲到来时进入寄存器，然后在其余数位进入和移位的时候，这个数据位从左到右移位。在 5 个时钟脉冲之后，此寄存器含有 $Q_4Q_3Q_2Q_1Q_0 = 11010$。参见图 9.6(b)。

相关问题[①]：如果该数据位反相，给出寄存器的状态。寄存器初始时清零。

8 位串行输入/串行输出移位寄存器的传统逻辑符号如图 9.7 所示。符号"SRG 8"表示具有 8 位容量的移位寄存器(SRG)。

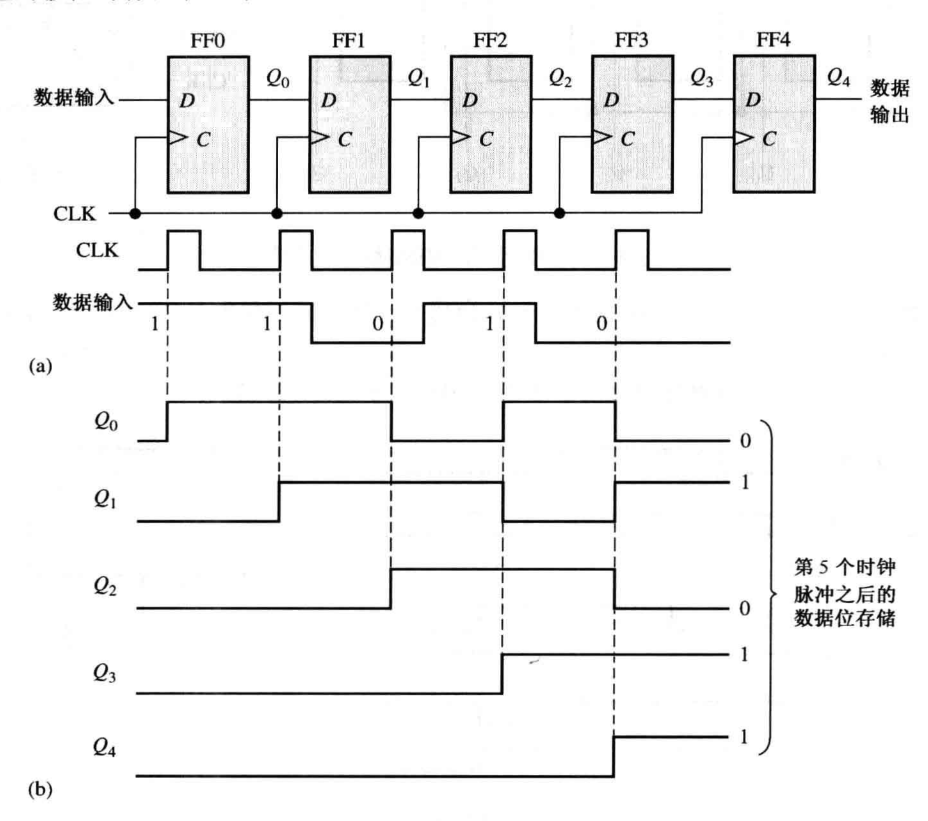


图 9.6　打开文件 F09-06 检验运行结果

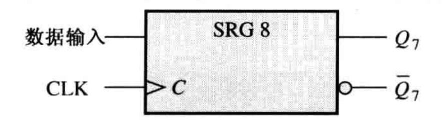


图 9.7　8 位串行输入/串行输出移位寄存器的逻辑符号

9.2 节　温故而知新

1. 使用 J-K 触发器替代 D 触发器，为图 9.3 的移位寄存器设计逻辑框图。
2. 串行输入一个字节到一个 8 位寄存器中，需要多少个时钟脉冲？

9.3　串行输入/并行输出移位寄存器

图 9.8 给出了一个 4 位串行输入/并行输出移位寄存器及它的逻辑方块符号。

---

① 相关问题的答案在本章的结尾。

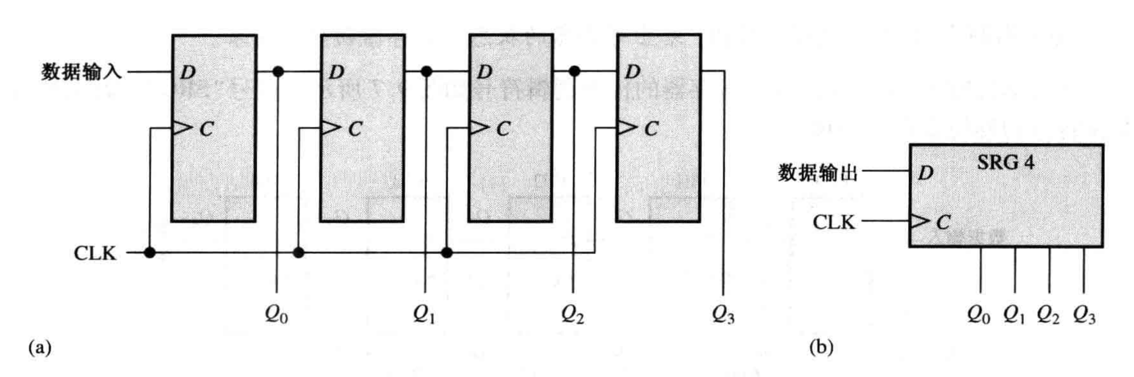


图 9.8　串行输入/并行输出移位寄存器

例 9.2　对于图 9.9(a)中的数据输入和时钟波形，给出此 4 位寄存器(SRG 4)的状态。寄存器初始时全为 1。

解： 该寄存器在 4 个时钟脉冲之后，状态为 0110，参见图 9.9(b)。

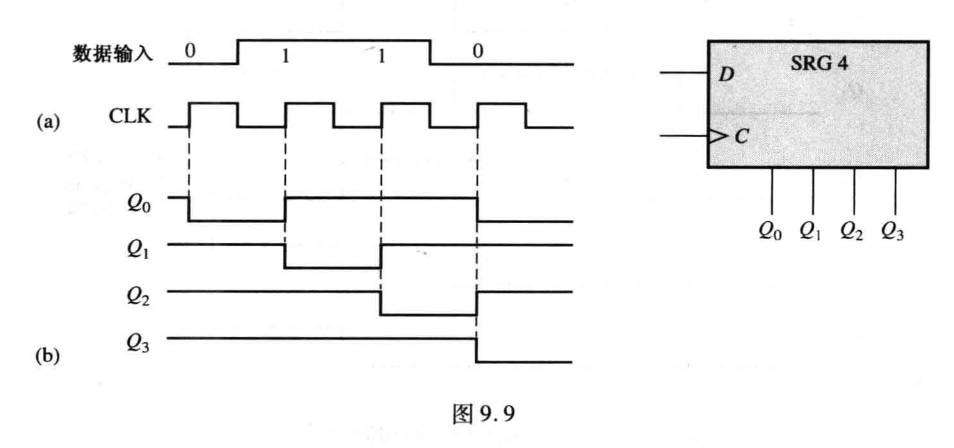


图 9.9

相关问题： 如果数据输入在第 4 个时钟脉冲之后仍然保持 0，那么在 3 个附加的时钟脉冲之后，寄存器的状态是什么？

9.3 节　温故而知新

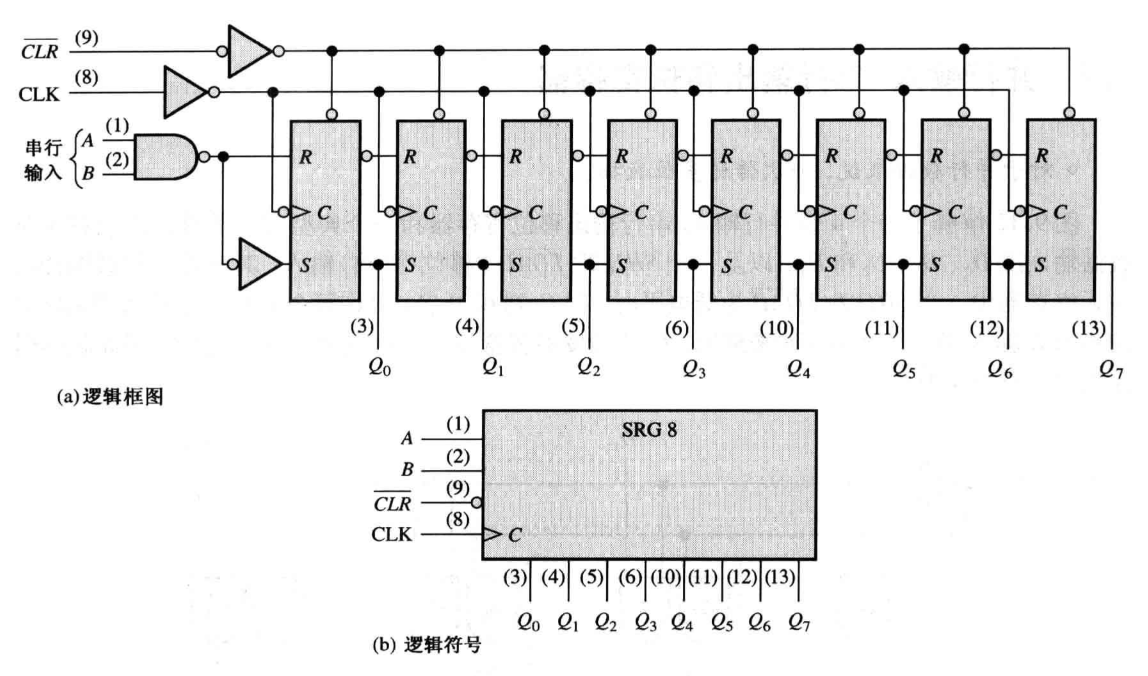
1. 位序列 1101 串行进入(从最低有效位开始)一个 4 位并行输出移位寄存器，初始状态为 0。两个时钟脉冲之后输出 Q 的状态是什么？
2. 一个串行输入/并行输出的寄存器可以用做串行输入/串行输出的寄存器吗？



74HC164 8 位串行输入/并行输出移位寄存器

74HC164 是具有串行输入/串行输出功能的 IC 移位寄存器的一个例子。逻辑框图如图 9.10(a)所示，典型的逻辑符号如图 9.10(b)所示。注意这个芯片具有两个门控串行输入 A 和 B，以及一个低电平有效的异步清零($\overline{CLR}$)输入。并行输出为 Q_0 到 Q_7。

74HC164 的一个样例时序图如图 9.11 所示。注意输入 A 上的串行输入数据在输入 B 变为高电平之后，移位并通过寄存器。



(a) 逻辑框图

(b) 逻辑符号

图 9.10　74HC164 8 位串行输入/并行输出移位寄存器

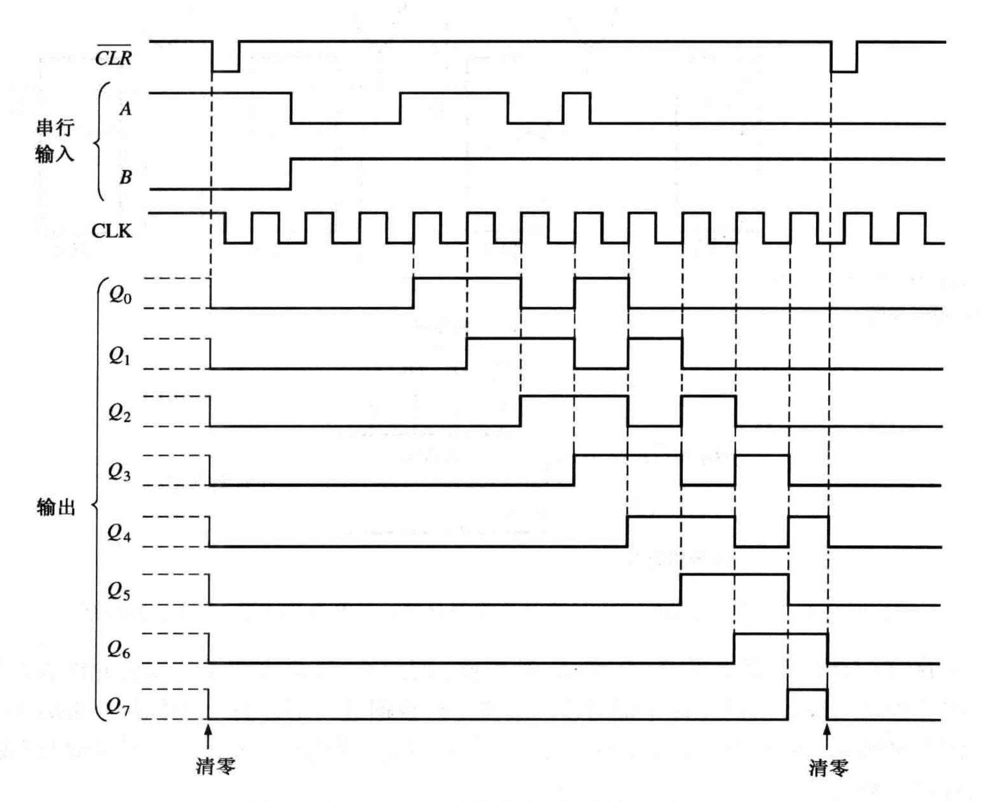


图 9.11　74HC164 移位寄存器的样例时序图

9.4 并行输入/串行输出移位寄存器

◇ 对于并行数据来说，一次传送多位数据。

图9.12解释了一个4位并行输入/串行输出移位寄存器和一个典型逻辑符号。注意有4条数据输入线D_0、D_1、D_2和D_3，以及一个$SHIFT/\overline{LOAD}$(移位/置数)输入，其允许4位数据并行进入寄存器中。当$SHIFT/\overline{LOAD}$为低电平时，门G_1到G_4开启，允许每个数据位分别加到相应触发器的D输入端。当时钟脉冲来到时，D_1的触发器将置位，而D_0的触发器将复位，因此就同时存储了所有的4个位。

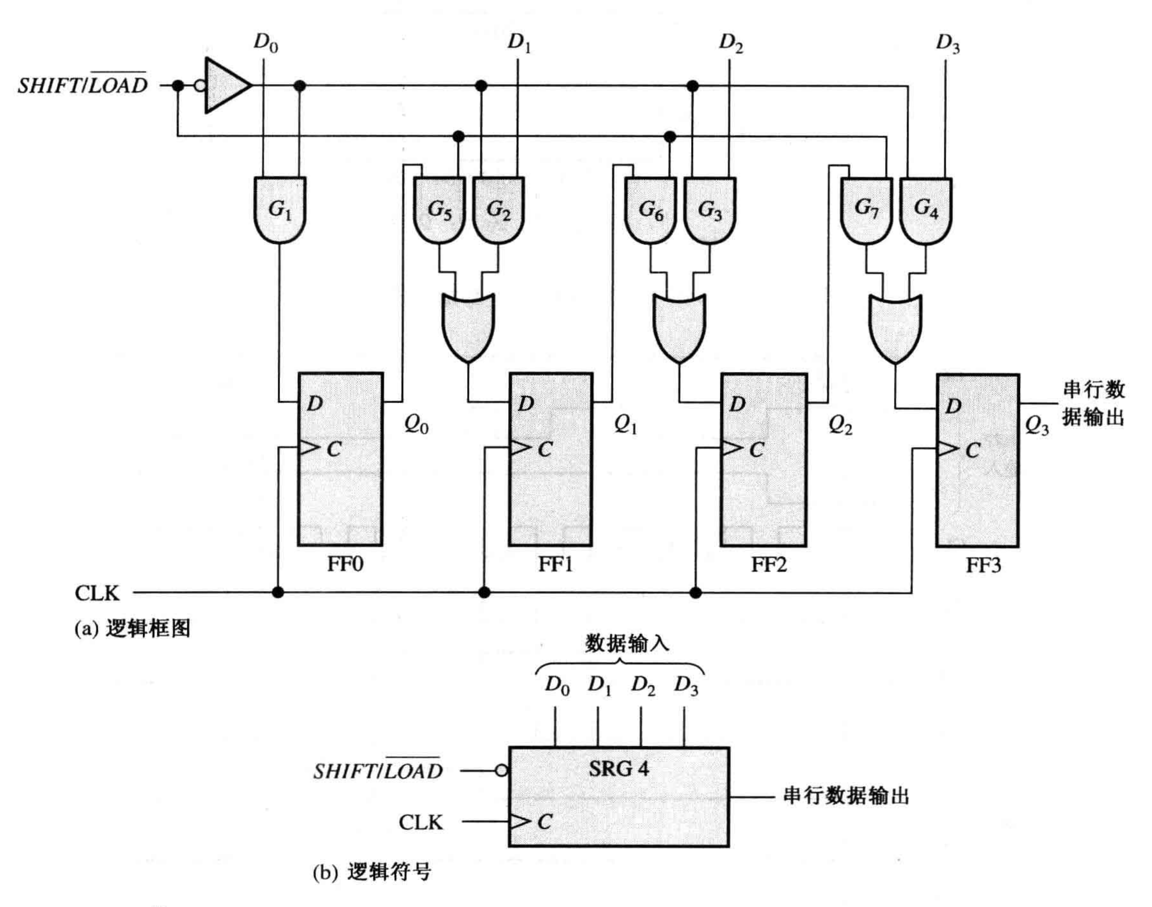


图9.12 4位并行输入/串行输出移位寄存器。打开文件F09-12检验运行结果

当$SHIFT/\overline{LOAD}$为高电平时，G_1到G_4被禁止，而门G_5到G_7被开启，这就允许数据位从一级向右移位到下一级。或门允许正常的移位或者并行数据进入的运算，这取决于$SHIFT/\overline{LOAD}$输入上的电平所开启的与门。注意，FF0有一个与门禁止并行输入D_0。这里不需要与/或连接，因为没有串行数据输入。

例9.3　对于图9.13(a)所示的并行输入数据、时钟及 $SHIFT/\overline{LOAD}$ 波形，给出4位寄存器的数据输出波形。参见图9.12(a)的逻辑框图。

解：时钟脉冲1到来时，并行数据($D_0D_1D_2D_3=1010$)被置入寄存器中，使得 Q_3 为0。时钟脉冲2到来时，来自 Q_2 的1被移位到 Q_3；时钟脉冲3到来时，0被移位到 Q_3；在时钟脉冲4，最后一个数据位(1)被移位到 Q_3；在时钟脉冲5上，所有的数据位已经被移出，只有1保存在寄存器中(假设 D_0 输入保持1)，参见图9.13(b)。

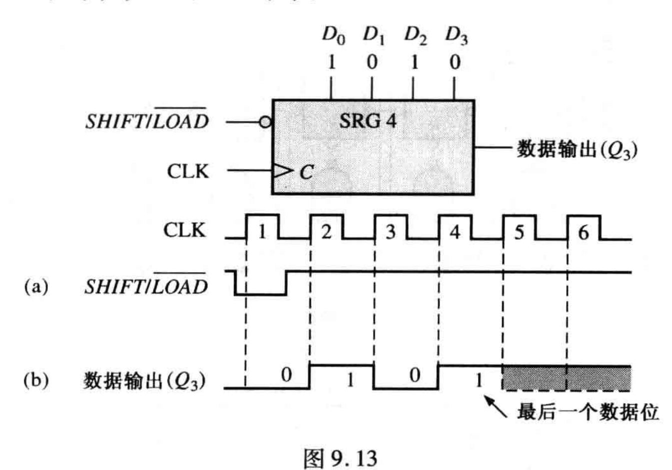
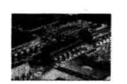


图9.13

相关问题：如果并行数据为 $D_0D_1D_2D_3=0101$，对于如图9.13(a)所示的时钟和 $SHIFT/\overline{LOAD}$ 输入，给出数据输出波形。



74HC165 8位并行置数移位寄存器

74HC165是具有并行输入/串行输出运算(也可以运行于串行输入/串行输出)的IC移位寄存器的一个例子。图9.14(a)给出了此芯片的内部逻辑框图，图9.14(b)给出了典型的逻辑符号。$SHIFT/\overline{LOAD}$($SH/\overline{LD}$)输入上的低电平开启并行置数的所有与非门。当输入数据位为1时，触发器就被上部的与非门的低电平异步置位。当输入数据位为0时，触发器被底部的与非门的低电平异步复位。此外，数据可以在串行(*SER*)输入上串行进入。同样，可以随时利用时钟禁止(*CLK INH*)输入上的高电平来禁止时钟输入。寄存器的串行数据输出为 Q_7 和它的反码 $\overline{Q}_7$。这种实现方法不同于前面所讨论的并行置数的同步方法，说明通常会有好几种方法来实现相同的功能。

图9.15是一个时序图，给出了74HC165移位寄存器的一个运算例子。

9.4节　温故而知新

1. 解释 $SHIFT/\overline{LOAD}$ (移位/$\overline{置数}$)输入的功能。
2. 在74HC165移位寄存器中的并行置数是同步的还是异步的？如何解释？

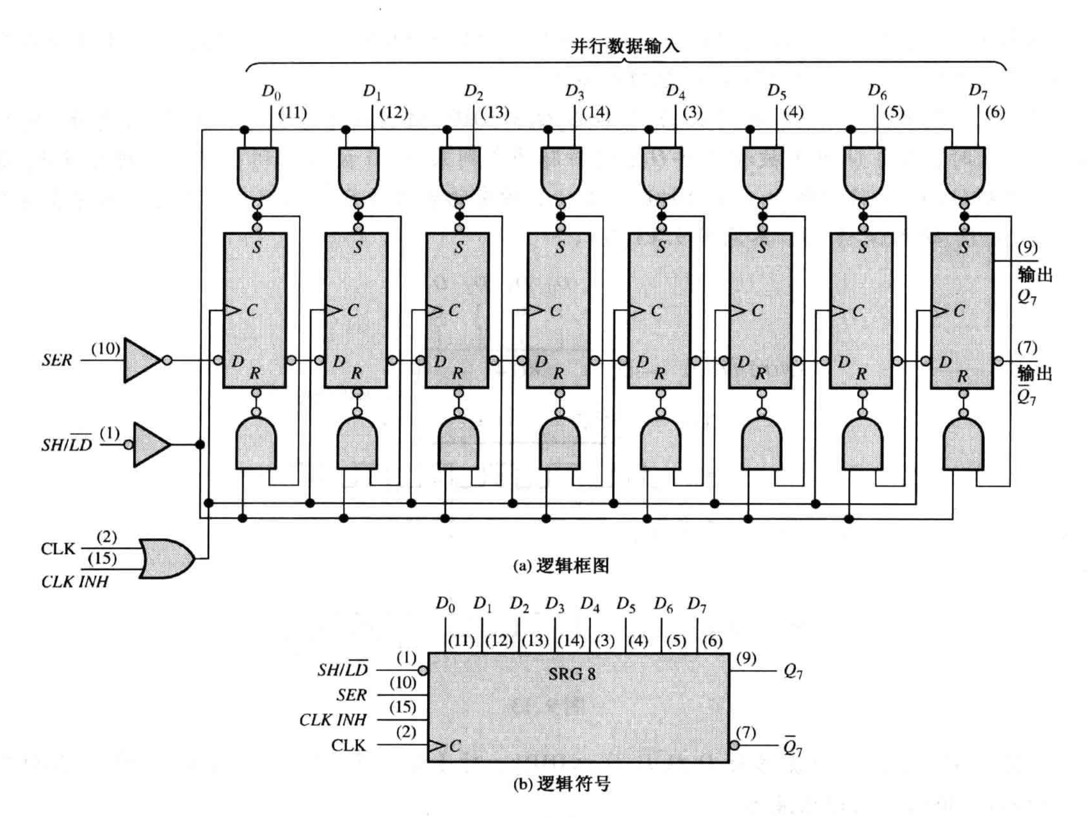


图 9.14　74HC165 8 位并行置数移位寄存器

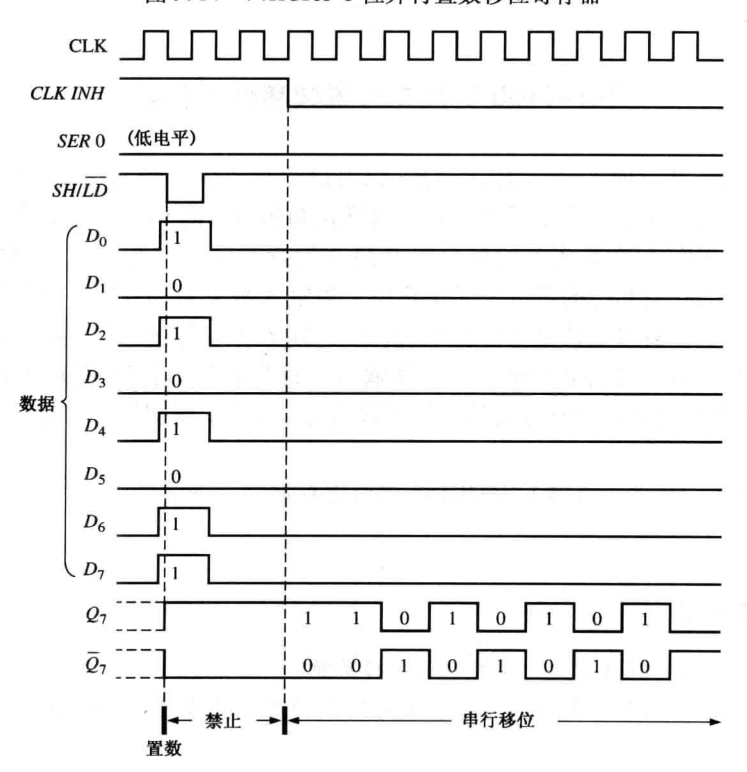


图 9.15　74HC165 移位寄存器的示例时序图

9.5 并行输入/并行输出移位寄存器

图9.16给出了一个并行输入/并行输出移位寄存器。

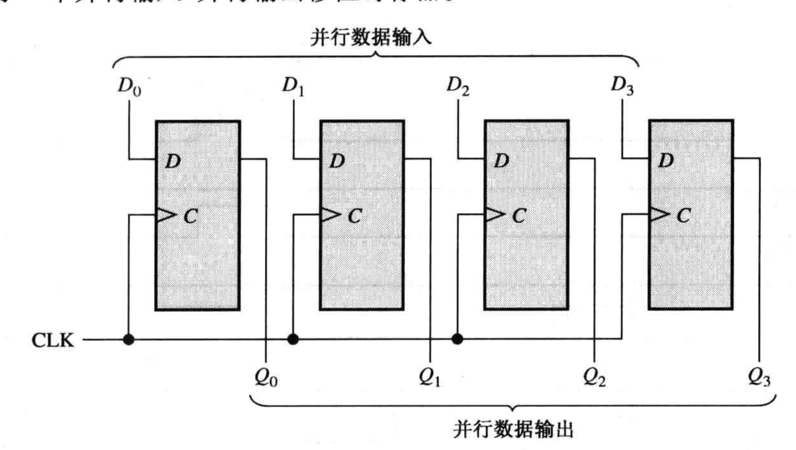


图9.16 并行输入/并行输出移位寄存器



74HC195 4位并行存取移位寄存器

74HC195可以用于并行输入/并行输出运算。因为它可以具有串行输入，所以也能用做串行输入/串行输出及串行输入/并行输出运算。通过把 Q_3 用做输出，它可以用于并行输入/串行输出运算。典型的逻辑框图符号如图9.17所示。

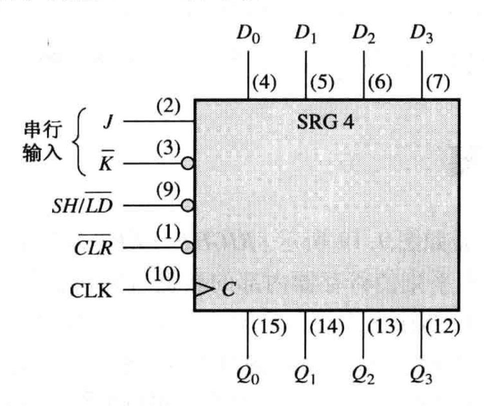


图9.17 74HC195 4位并行存取移位寄存器

当 $SHIFT/\overline{LOAD}$ 输入（$SH/\overline{LD}$）为低电平时，并行输入上的数据在时钟的上升沿转换到来时同步进入。当 $SH/\overline{LD}$ 为高电平时，存储的数据将会随着时钟向右同步移位（Q_0 到 Q_3）。输入 J 和 $\overline{K}$ 是寄存器第一级（Q_0）的串行数据输入；Q_3 可以用做串行输出数据。低电平有效清零输入是同步的。

图9.18中的时序图解释了此寄存器的运算。

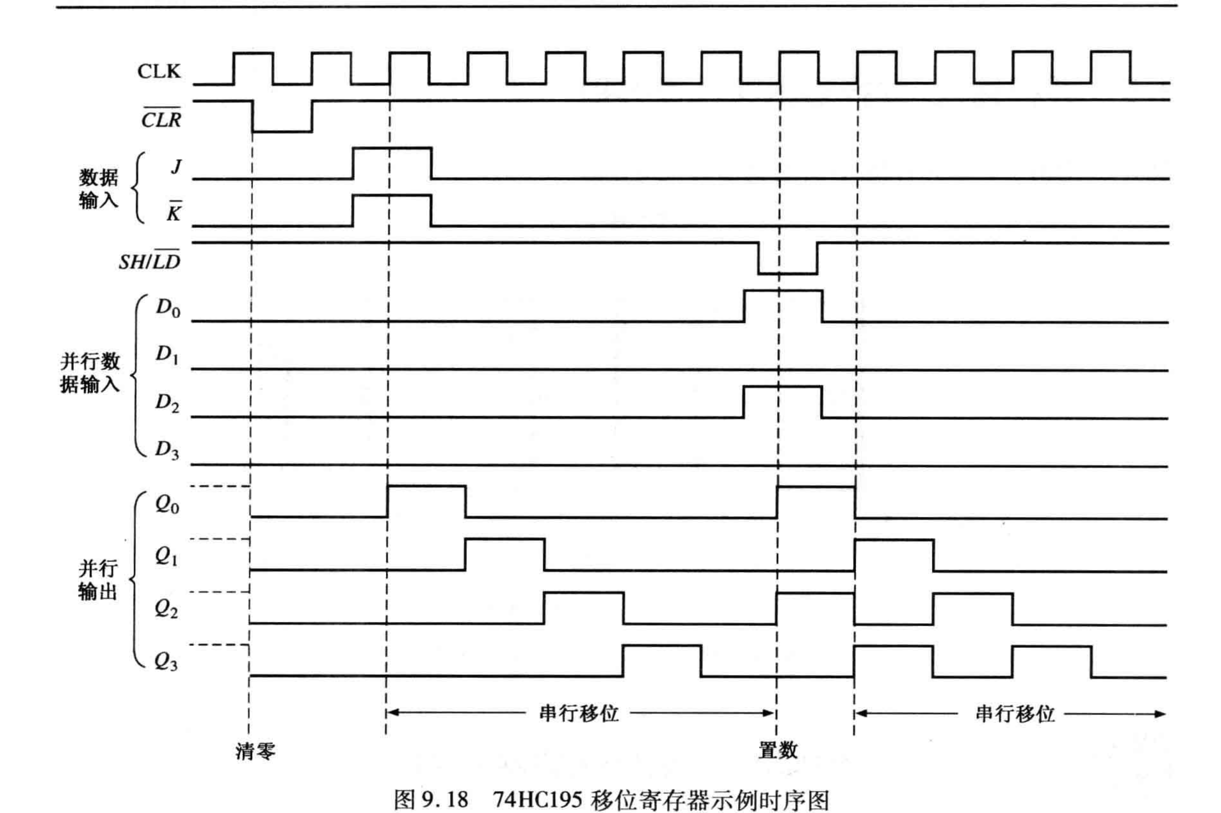


图 9.18 74HC195 移位寄存器示例时序图

9.5 节 温故而知新

1. 在图 9.16 中，$D_0=1$、$D_1=0$、$D_2=0$ 和 $D_3=1$。3 个时钟脉冲之后，数据输出是什么？

2. 74HC195 芯片的 $SH/\overline{LD}=1$，$J=1$ 和 $\overline{K}=1$。1 个时钟脉冲之后，Q_0 是什么？

9.6 双向移位寄存器

一个 4 位双向移位寄存器如图 9.19 所示。$RIGHT/\overline{LEFT}$ 控制输入上的高电平允许寄存器内部的数据位向右移位，而低电平则使寄存器内部的数据位向左移位。门控逻辑的检查将使运算显而易见。当 $RIGHT/\overline{LEFT}$ 控制输入为高电平时，门 G_1 到 G_4 开启，而每个触发器的 Q 输出的状态传送到下一个触发器的 D 输入上。当时钟脉冲来到时，数据位向右移动一个位置。当 $RIGHT/\overline{LEFT}$ 控制输入为低电平时，门 G_5 到 G_8 开启，而每个触发器的 Q 输出传送到前一个触发器的 D 输入上。当时钟脉冲到来时，数据位就会向左移动一个位置。

例 9.4 对于图 9.20(a) 中给定的 $RIGHT/\overline{LEFT}$ 控制输入波形，确定图 9.19 中移位寄存器在每个时钟脉冲之后的状态。假设 $Q_0=1$、$Q_1=1$、$Q_2=0$ 和 $Q_3=1$，并且串行数据输入线为低电平。

解： 参见图 9.20(b)。

相关问题： 把 $RIGHT/\overline{LEFT}$ 波形反相，确定图 9.19 中移位寄存器在每个时钟脉冲之后的状态。

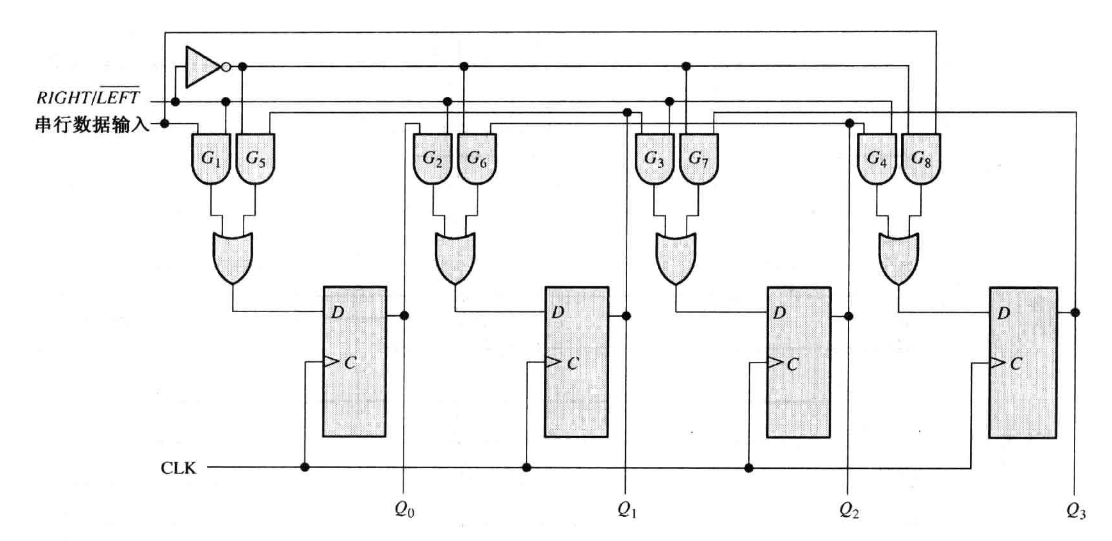


图 9.19　4 位双向移位寄存器。打开文件 F09-19 检验运行结果

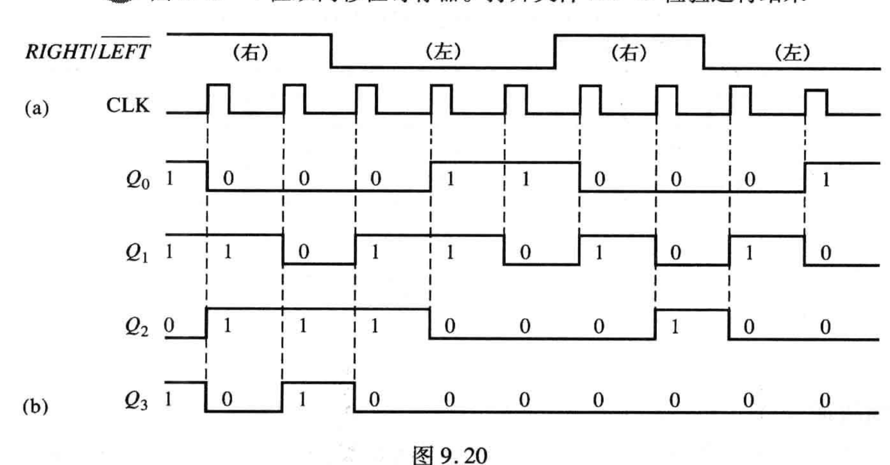


图 9.20



74HC194 4 位通用双向移位寄存器

74HC194 是集成电路形式中通用双向移位寄存器的一个例子。通用移位寄存器同时具有串行和并行输入与输出的能力。逻辑符号如图 9.21 所示，而示例时序图如图 9.22 所示。

同步于时钟上升沿转换的并行置数，是通过将 4 个数据位加到并行输入及将一个高电平加到 S_0 和 S_1 输入上来实现的。当 S_0 为高电平和 S_1 为低电平时，向右移位和时钟的上升沿同步完成。此模式的串行数据在向右移位串行输入(*SR SER*)上进入寄存器。当 S_0 为低电平和 S_1 为高电平时，数据位左移同步于时钟，新数据左移串行(*SL SER*)进入寄存器。输入 *SR SER* 进入 Q_0 触发器，而 *SL SER* 进入 Q_3 触发器。

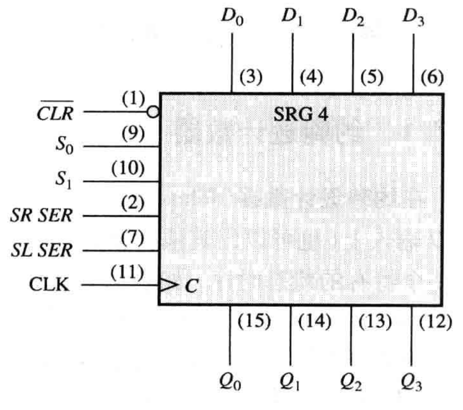


图 9.21　74HC194 4 位通用双向移位寄存器

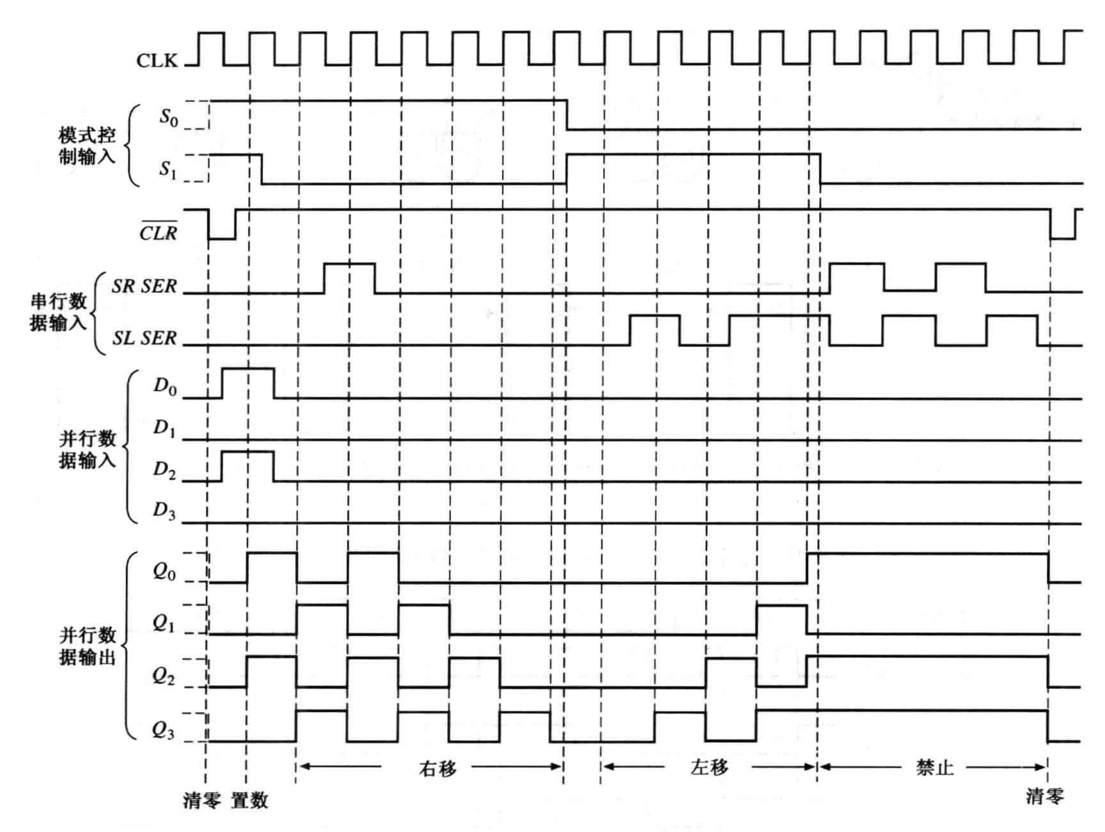


图 9.22　74HC194 移位寄存器的示例时序图

9.6 节　温故而知新

1. 假设图 9.19 的 4 位双向移位寄存器有如下的数据：$Q_0 = 1$，$Q_1 = 1$，$Q_2 = 0$，$Q_3 = 0$。串行数据输入线上为 1。如果是 3 个时钟脉冲对应于 $RIGHT/\overline{LEFT}$ 为高电平，两个对应于低电平，那么 5 个时钟脉冲以后的数据是什么？

9.7　移位寄存器计数器

9.7.1　约翰逊计数器

在**约翰逊计数器**(Johnson counter)中，最后一个触发器输出的反码连接回到第一个触发器的 D 输入上(也可以用其他类型的触发器来实现)。如果计数器开始于零状态，这种反馈连接产生一个特有的状态时序，如表 9.1 所示的 4 位约翰逊计数器及表 9.2 所示的 5 位约翰逊计数器。注意，4 位序列总共具有 8 个状态或位模式，5 位序列总共具有 10 个状态。一般来说，约翰逊计数器将会产生模 $2n$，其中 n 是计数器中级的个数。

4 位和 5 位约翰逊计数器的实现方法如图 9.23 所示。约翰逊计数器的实现是非常直接的，除了级数不同之外都是一样的。每一级的 Q 输出都连接到下一级的 D 输入上(假设使用 D 触发

器)。唯一的例外是最后一级的 $\bar{Q}$ 输出连接回到第一级的 D 输入上。如表 9.1 和表 9.2 所给出的时序那样，计数器将从左到右"填充"1，然后再"填充"0。

4 位和 5 位约翰逊计数器的时序运算图分别如图 9.24 和图 9.25 所示。

表 9.1　4 位约翰逊计数器的时序

时钟脉冲	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

表 9.2　5 位约翰逊计数器的时序

时钟脉冲	Q_0	Q_1	Q_2	Q_3	Q_4
0	0	0	0	0	0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	1	0
5	1	1	1	1	1
6	0	1	1	1	1
7	0	0	1	1	1
8	0	0	0	1	1
9	0	0	0	0	1

(a) 4 位约翰逊计数器

(b) 5 位约翰逊计数器

图 9.23　4 位和 5 位约翰逊计数器

图 9.24　4 位约翰逊计数器的时序图

图 9.25　5 位约翰逊计数器的时序图

9.7.2 环形计数器

环形计数器为时序中的每个状态都使用一个触发器。它具有的优点是不需要译码门。在10位环形计数器的情况下，每个十进制数都有唯一的一个输出。

10位环形计数器的逻辑框图如图9.26所示。环形计数器的时序如表9.3所示。初始时，一个1被预置到第一个触发器中，而其余的触发器都被清零。注意级间连接和约翰逊计数器是一样的，不同的是Q而不是$\bar{Q}$从最后一级中反馈回去。计数器的10个输出直接指明了时钟脉冲的十进制计数。例如，Q_0上的1表示0，Q_1上的1表示1，Q_2上的1表示2，而Q_3上的1表示3，以此类推。应当验证1总是保持在计数器中，并且简单地“围绕圆环”移位，每个时钟脉冲到来时移动一级。

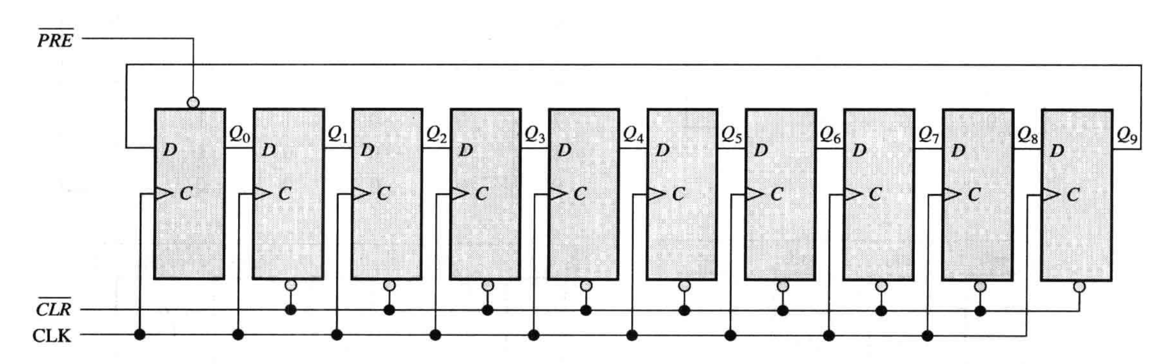


图9.26 10位环形计数器。打开文件F09-26检验运行结果

表9.3 10位环形计数器时序

时钟脉冲	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	0	1

使计数器中具有多于一个的单个1，就可以得到修改的时序，如例9.5所示。

例9.5 如果和图9.26相似的一个10位环形计数器的初始状态为1010000000，确定每个Q输出的波形。

解： 参见图9.27。

相关问题： 如果一个10位环形计数器具有初始状态0101001111，确定每个Q输出的波形。

9.7节 温故而知新

1. 在8位约翰逊计数器中，有多少种状态？
2. 开始于000状态，写出3位约翰逊计数器。

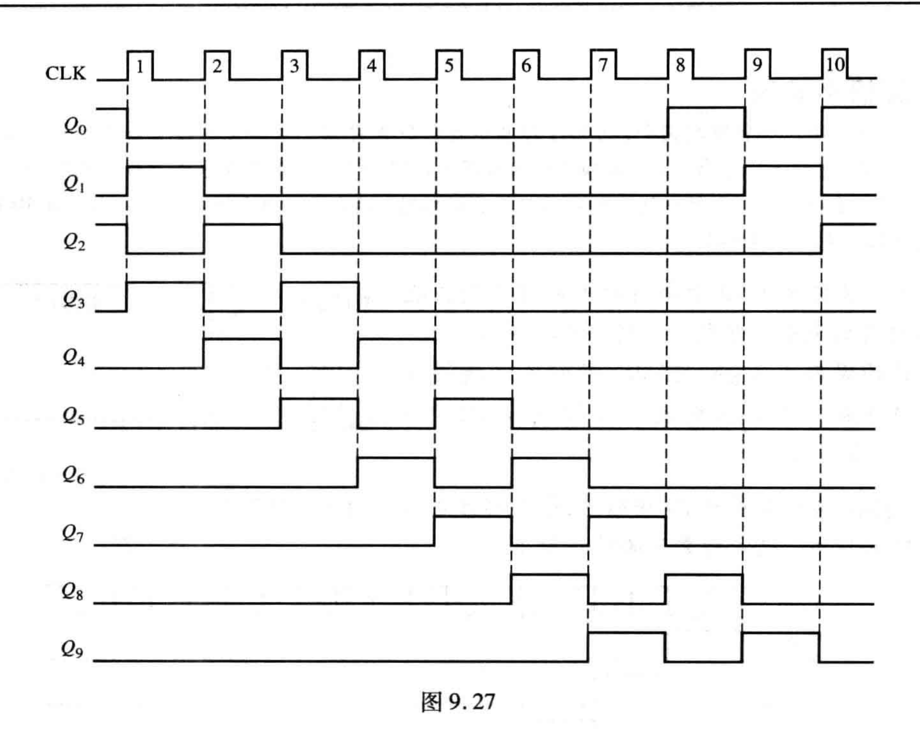


图 9.27

9.8　移位寄存器应用

9.8.1　时间延迟

串行输入/串行输出移位寄存器可以用来提供从输入到输出的时间延迟，此时间延迟是寄存器中级数(n)和时钟频率的函数。

当一个数据脉冲加在串行输入时，如图 9.28 所示(A 和 B 连接在一起)，它就在时钟脉冲的触发边沿进入第一级。然后在每一个相继的时钟脉冲作用下，从一级移位到下一级，直至在 n 个时钟周期之后出现在串行输出上。这个时间延迟运算如图 9.28 所示，其中使用了一个具有 1 MHz时钟频率的 8 位串行输入/串行输出移位寄存器，从而实现 8 μs 的时间延迟 t_d(8 × 1 μs)。通过改变时钟频率可以增加或者减少这个时间。时间延迟也可以通过级联移位寄存器来增加，或者通过从连续的较低级处取出输出来减少时间延迟，前提是输出可用，如例 9.6 所示。

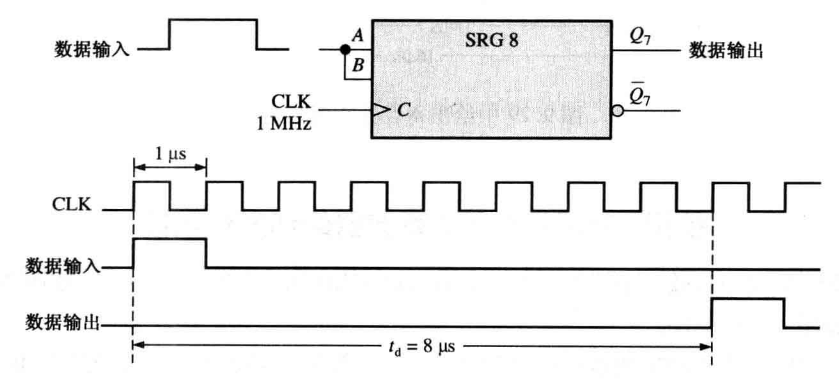


图 9.28　移位寄存器作为时间延迟芯片



计算机小知识

微处理器都具有一些特殊的指令，可用于仿真串行移位寄存器。累加寄存器可以左移或右移数据。右移等价于除以2，左移等价于乘以2。累加器中的数据可以用循环指令实现左移或右移。ROR 为循环右移指令，ROL 为循环左移指令。当循环指令运行后有多出的位时，两个其他的指令将处理进位，即 RCR(循环右移进位)及 RCL(循环左移进位)。

例 9.6 确定图 9.29 中串行输入和每个输出之间的时间延迟的大小。绘制一个时序图来说明。

解： 时钟周期是 2 μs。因此，时间延迟能够以 2 μs的增量在最小值 2 μs 到最大值 16 μs 之间增加或者减少，如图 9.30 所示。

相关问题： 要得到图 9.29 中输入到 Q_7输出之间的 24 μs 时间延迟，确定所需要的时钟频率。

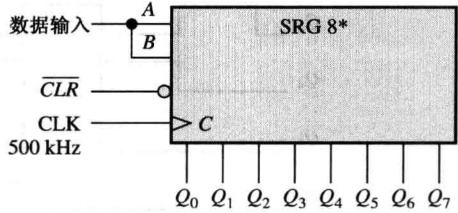


图 9.29

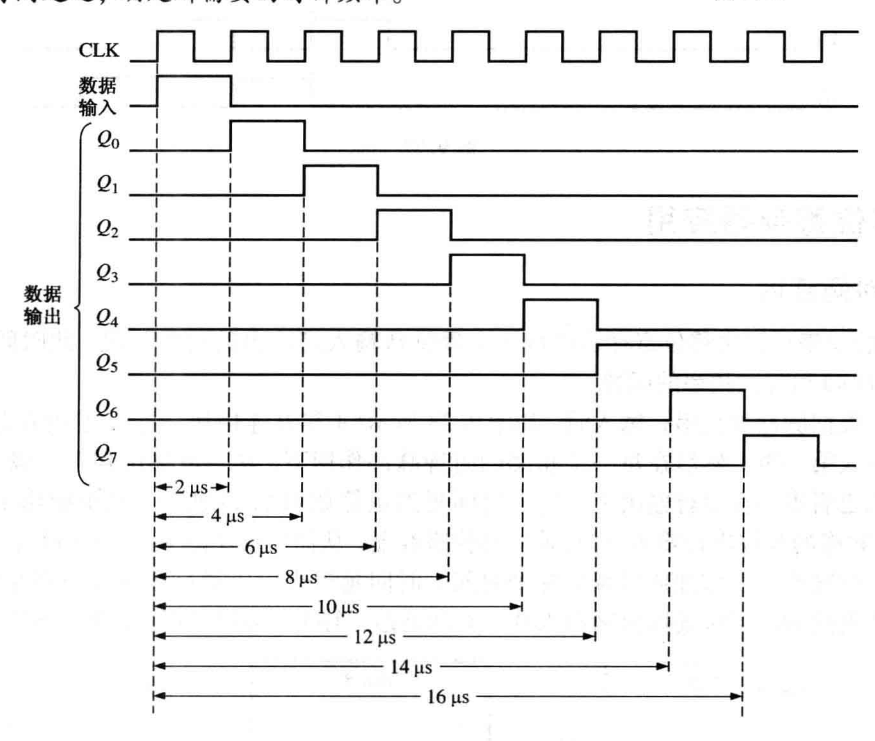


图 9.30 图 9.29 中给出寄存器时间延迟的时序图



使用 74HC195 移位寄存器的环形计数器

如果输出回接到串行输入上，移位寄存器就可以用做环形计数器。图 9.31 使用 74HC195 4 位移位寄存器解释了这种应用。

初始时，位模式为 1000(或者任何其他组合)可以被同步预置数到计数器中，即将此位模式应用到并行数据输入上，使 $SH/\overline{LD}$ 输入为低电平，同时加上一个时钟脉冲。在这个初始化之后，继续在环形计数器中循环，如图 9.32 的时序图所示。

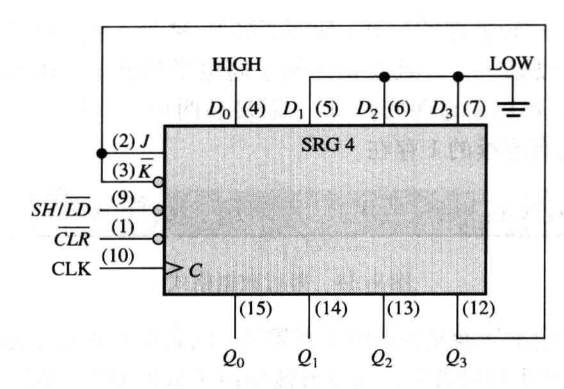


图9.31　连接为环形计数器的74HC195

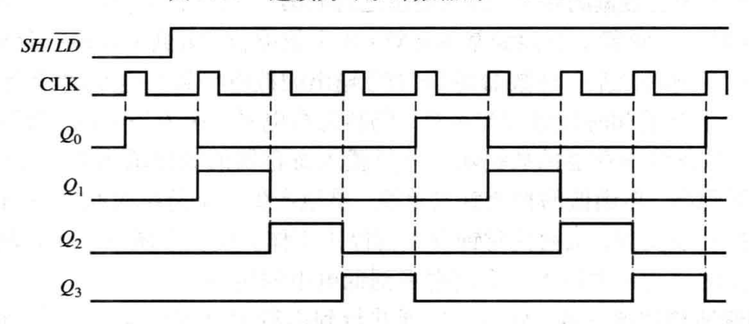


图9.32　给出图9.31中环形计数器的两个完整循环的时序图，初始时预置数1000

9.8.2　串行到并行数据转换器

从一个数字系统到另一个数字系统的串行数据传送，常常用来减少传输线中的导线数。例如，8个位可以在一条导线上串行传送，但是并行传送同样的数据则需要8条导线。

串行传送广泛地应用在外围设备把数据传送到计算机和从计算机接收数据。例如，用来连接键盘、打印机、扫描仪和其他更多设备到计算机的USB(通用串行总线)。所有的计算机以并行的方式处理数据，因此需要串行到并行的转换。一个简单的串行到并行的数据转换如图9.33所示，它使用两种类型的移位寄存器。

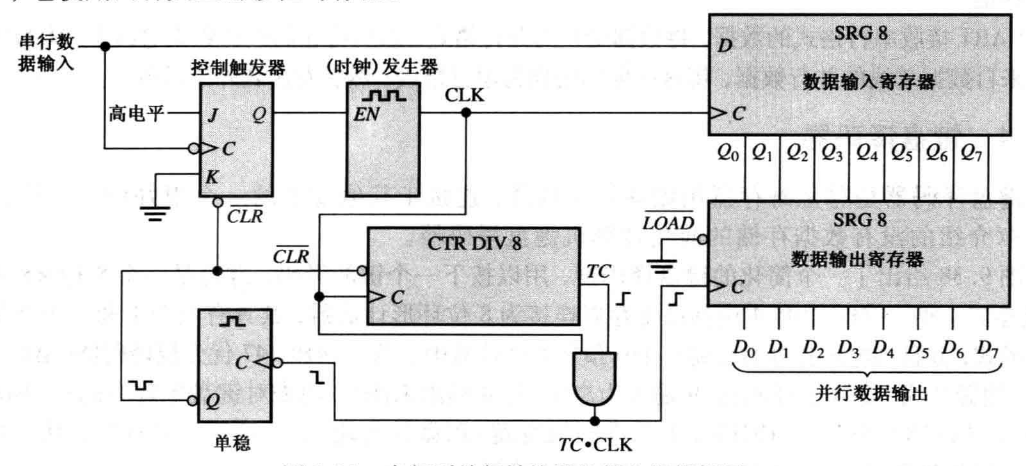


图9.33　串行到并行转换器的简化逻辑框图

为了解释此串行到并行转换器的运算，使用了图9.34所示的串行数据格式。它由11个位组成。第一个位(起始位)总是0，并且总是开始于高电平到低电平的转换。下面的8位(D_7到D_0)都是数据位(其中一位可以是奇偶校验位)，而最后两位(终止位)总是1。当没有数据被传送时，串行数据线上就会有连续的1存在。

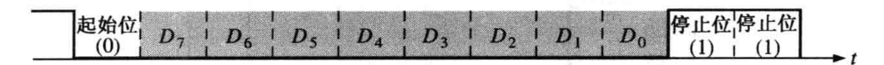


图9.34 串行数据格式

起始位的高电平到低电平转换使控制触发器置位，以启动时钟发生器。在一个固定的延迟时间之后，时钟发生器开始产生脉冲波形，此波形被应用于数据输入寄存器和8分频计数器。时钟的频率精确等于输入串行数据的频率，而起始位之后的第一个时钟脉冲在第一个数据位期间发生。

图9.35中的时序图解释了下面的基本运算：8个数据位(D_7到D_0)串行移位到数据输入寄存器。在第8个时钟脉冲之后，计数器终端(*TC*)输出的低电平到高电平的转换，表示计数器进入最后状态。这个上升沿和时钟进行与运算，仍然为高电平，在 *TC*·CLK 的输出处产生一个上升沿。8个数据位从移位寄存器的数据输入并行置入寄存器的数据输出端。很短的一段时间之后，时钟脉冲从低到高、再到低的瞬变触发单稳，单稳产生一个持续时间很短的脉冲，将计数器清零，并且复位控制触发器，从而使时钟发生器停止工作。现在系统已准备好转换下一组11位数，并且在起始位的开始处等待下一个高电平到低电平的转换。

通过反转刚刚所描述的过程，就可以实现并行到串行数据的转换。但是，由于必须产生串行数据格式，所以必须在数据序列中添加开始位和结束位。

9.8.3 通用异步接收发送机(UART)

正如提到的那样，基于计算机和微处理器的系统常常发送和接收并行格式的数据。在许多情况下，这些系统必须和发送和/或接收串行数据的外部设备进行通信。用来完成这些转换的接口设备是UART(Universal Asynchronous Receiver Transmitter，通用异步接收发送机)。图9.36解释了基于一般微处理器的系统应用中的UART。

UART包括一个串行到并行和并行到串行的转换器，如图9.37所示。数据总线基本上是一组并行导线，数据沿着这些导线在UART和微处理器系统之间移动。缓冲器连接数据总线和数据寄存器。

UART接收串行格式的数据，将数据变换为并行格式，然后将它们放在数据总线上。UART还接收来自数据总线的并行数据，将这些数据变换为串行格式，然后发送到外围设备上。

9.8.4 键盘译码器

键盘译码器是移位寄存器用做环形计数器，连接于其他设备的一个很好例子。回顾在第6章介绍的没有数据存储的简化计算机键盘译码器。

图9.38给出了一个简化的键盘译码器，用以按下一个键的译码，键盘是一个8行×8列的64键矩阵。两个74HC195 4位移位寄存器连接为8位环形计数器，其具有七个1和一个0的固定位模式，并且当电源打开时会将一个0预置到计数器中。两个74HC147优先权译码器(第6章介绍过)用做8线-3线的译码器(9输入为高电平，8输出未用)，用来对键盘矩阵中的行(ROW)和列(COLUMN)译码。74HC174(十六进制触发器)用做并行输入/并行输出寄存器，其中存储了来自优先译码器的行(ROW)/列(COLUMN)代码。

图9.35　图9.33中串行到并行数据转换器运算的时序图说明

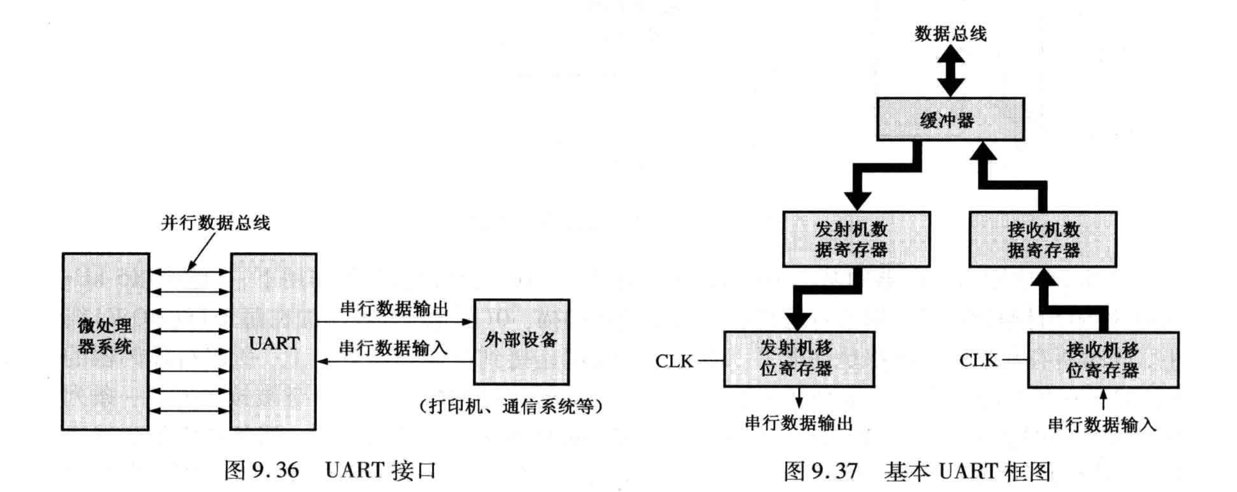


图9.36　UART接口

图9.37　基本UART框图

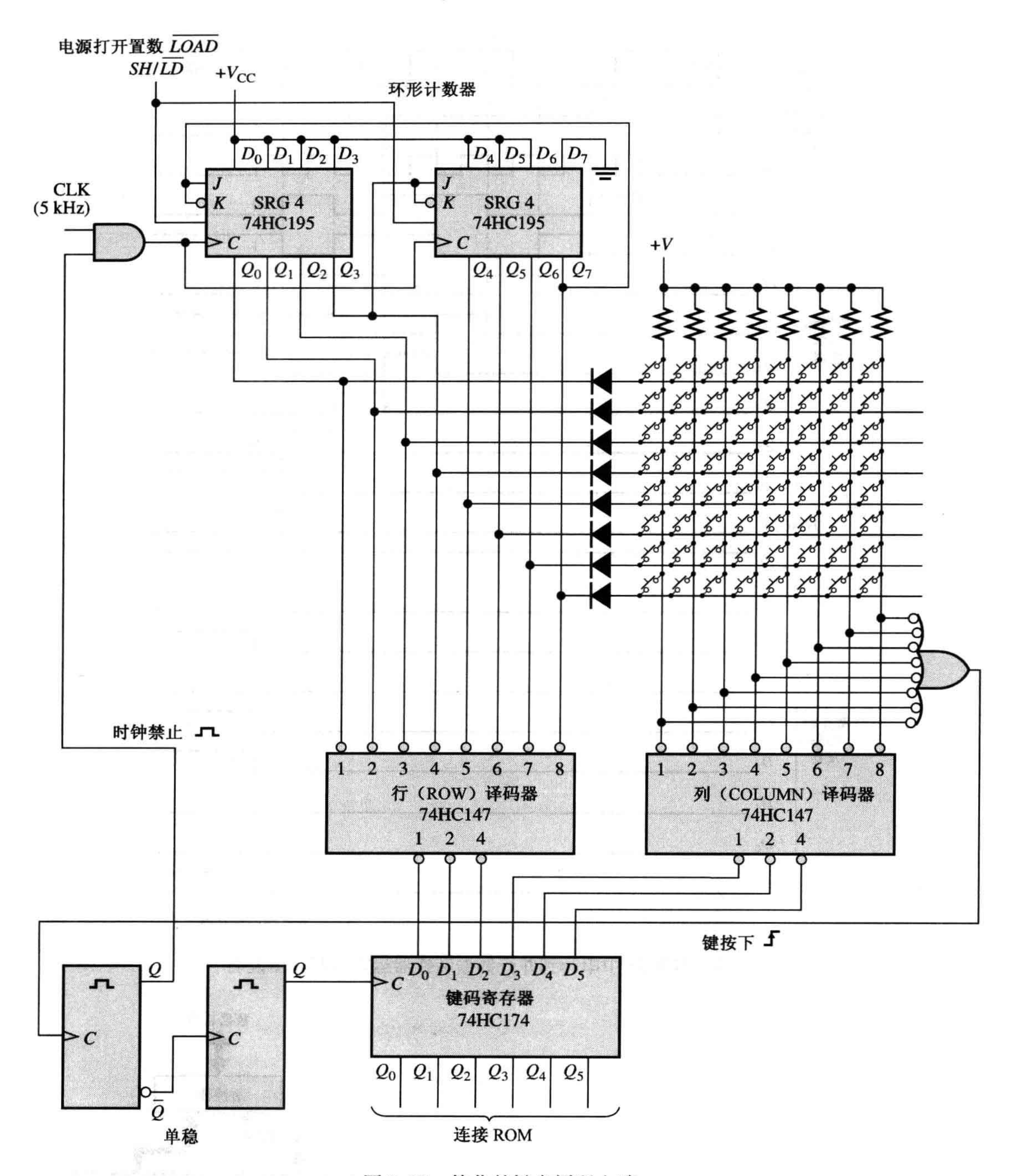


图 9.38　简化的键盘译码电路

图 9.38 中键盘译码器的基本运算如下所示：在 5 kHz 的时钟信号的作用下，0 输出以5 kHz 的速率围绕计数器移位，用来对是否有键按下进行扫描。0(低电平)顺序加在每个行(ROW)线上，同时所有其他的行线都是高电平。所有的行线都连接到行译码器输入上，所以行译码器的 3 位输出在任何时候都是低电平状态下的行线所表示的二进制数。当一个键被按下时，一条列(COLUMN)线就和一条行线连接。当这条行线被环形计数器置为低电平时，那条特殊的列线也同样被置为低电平。列译码器产生一个二进制输出，这个二进制输出和键被按下的那一列相对

应。3位行代码加上3位列代码就唯一确定了被按下的键。此6位代码加到键码寄存器的输入。当一个键被按下时，两个单稳就会产生一个延迟的时钟脉冲，把6位代码并行置入键码寄存器中。这个延迟可以消除按键触点的抖动。同样，第一个单稳输出使环形计数器停止工作，这样在数据置入键码寄存器的时候扫描停止。

键码寄存器中的6位代码现在加到ROM(只读存储器)，以变换为识别键盘字符的相应的字母数字代码。在第10章将介绍ROM。

9.8节 温故而知新

1. 在键盘译码器中，每秒时间里环形计数器对键盘扫描多少次？
2. 在键盘译码器中，顶部的行和最左列处的6位行/列码(键码)是什么？
3. 在键盘译码器中二极管的用途是什么？电阻的用途是什么？

9.9 关联标注的逻辑符号

74HC164 8位串行输入/并行输出移位寄存器的逻辑符号如图9.39所示。共用的控制输入在有凹槽的方块中给出。清零($\overline{CLR}$)输入由方块内的R(用于复位)来表示。由于没有关联前缀连接R和时钟(C1)，所以清零功能是异步的。C1后面的右箭头符号表示数据从Q_0流到Q_7。A和B输入相与，由内置与“&”符号表示，用以提供同步数据输入，1D到第一级(Q_0)。注意D关联于C，由C的1后缀和D的1前缀来表示。

图9.40是74HC194 4位双向通用移位寄存器的逻辑符号。开始于控制方块的左上角，注意$\overline{CLR}$输入是低电平有效并且是异步的(没有前缀和C连接)。输入S_0和S_1是模式输入，确定运算的右移、左移和并行置数模式，由M后面的0/3关联符号来表示。0/3表示S_0和S_1输入上的0、1、2和3这些二进制状态。当这些数字的某一个用做另一个输入的前缀时，就建立了关联。

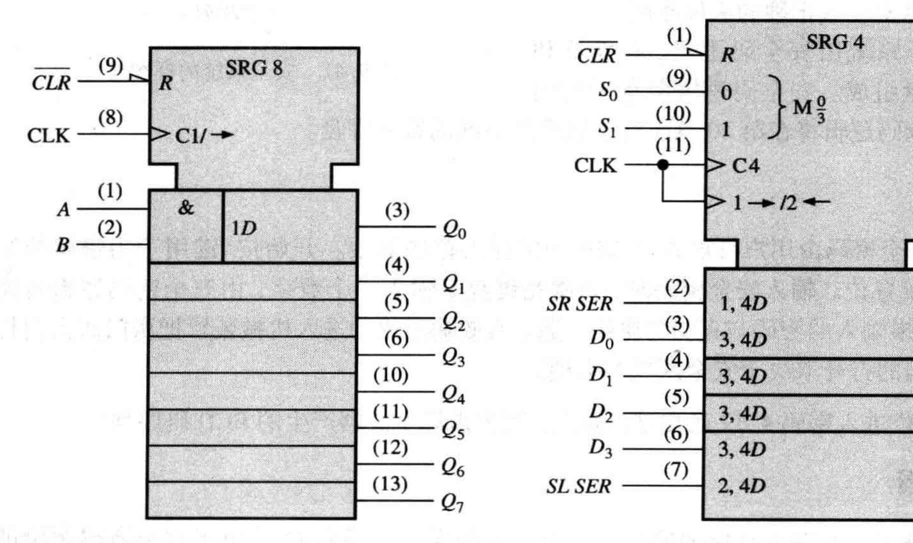


图9.39 74HC164的逻辑符号

图9.40 74HC194的逻辑符号

时钟输入上的1→/2←符号表示下面的内容：1→表示当模式输入(S_0，S_1)处于二进制1状态($S_0=1$，$S_1=0$)时，向右移位(Q_0向Q_3)，2←表示当模式输入处于二进制2状态($S_0=0$，$S_1=1$)时，向左移位(Q_3向Q_0)。右移串行输入($SR\ SER$)为模式关联和时钟关联，由“1，4D”来表示。

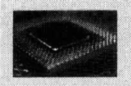
并行输入(D_0、D_1、D_2和D_3)都是模式关联(前缀3表示并行置数模式)和时钟关联，由“3，4D”来表示。左移串行输入(*SL SER*)为模式关联和时钟关联，由“2，4D”来表示。

74HC194 的 4 种模式总结如下：

不做任何事情：$S_0=0$，$S_1=0$(模式0)
右移：$S_0=1$，$S_1=0$(模式1，如“1，4D”)
向左移位：$S_0=0$，$S_1=1$(模式2，如“2，4D”)
并行置数：$S_0=1$，$S_1=1$(模式3，如“3，4D”)

9.9 节　温故而知新

1. 在图9.40中，是否有任何输入依赖于0状态中的模式输入？
2. 并行置数是否同步于时钟？



数字系统应用

安全系统：第一部分

在这个数字系统应用中，将要设计一个安全系统，即提供进入安全区域的安全密码。一旦将4位数字安全密码存储在系统中，就必须在键盘上输入正确的安全密码才能进入。图9.41给出了安全系统的框图。

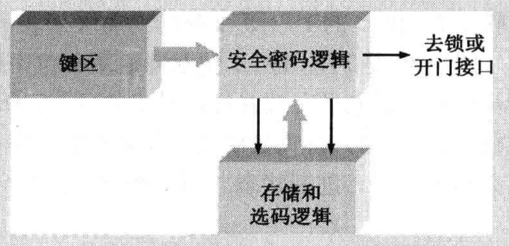


图9.41　安全系统的框图

在解除(防卫)模式下，通过一次性从小键盘上得到的4位数的安全密码，系统就可以对此码进行编程。一旦输入了安全密码并且存储起来，系统就进入防卫模式。要想解除防卫，就必须在小键盘上键入正确的4位密码。

如图所示，系统由安全码逻辑、存储器和选码逻辑、键盘组成。安全码逻辑着重于应用实践，存储和选码逻辑将在第10章设计。键盘是标准的数字键盘。

基本运算

4位进入安全密码由用户可输入的DIP开关存入存储器中。开始按#键用于为密码的第一个数字对系统置位。输入安全密码时，每次在键盘上输入一个数字，由安全密码逻辑转换为BCD码。如果输入的密码与存储的密码一致，有效输出驱动进入机械装置把房门或大门打开，房门或大门的打开取决于安全区域的类型。

1. 如果4位进入密码4739已在键盘输入，写出由码生成器产生的BCD码序列。

安全密码逻辑

安全密码逻辑比较预先存储的密码和键盘输入的密码。图9.42给出了安全密码逻辑的逻辑框图。

为了进入门，首先按下键盘上的#键触发单稳，从而以预置00010000的模式初始化8位寄存器C。下一步，在键盘上输入正确序列的4位密码。当每位密码输入之后，它被十-二进制

译码器转换成BCD码，由单稳A产生一个时钟脉冲式的4位码移位进入寄存器A。当键被按下时，或门输出的瞬间变换触发单稳。与此同时，来自码发生器的对应数据移入寄存器B。单稳A输出一个延迟的时钟脉冲到寄存器C，把预置数(00010000)移位，然后触发单稳B。最左边的3个0只是简单地填充数据，在系统的运算中没有用途。寄存器A和B的输出加到比较器上；如果两个码相同，比较器的输出为高电平，使得移位寄存器C进入移位模式。

每次输入的数字与预置的时钟一致时，移位寄存器C中的1向右移动一个位置。第4个数字一致时，移位寄存器输出为1，驱动机械装置开锁或开门。如果输入的数字和预置的不一致，比较器输出低电平，使得移位寄存器C处在置数模式，寄存器就重新初始化进入预置数模式(00010000)。

2. 输入两个正确的数字以后，移位寄存器C的状态是什么？
3. 解释或门的用途。
4. 如果在键盘上输入数字4，在寄存器A的输出是什么？

计算机仿真

图9.43给出了安全密码逻辑的计算机仿真界面。一个DIP开关用于模拟10个数字的键盘，开关J1模拟#键。开关J2～J5用于测试的用途，输入在整个系统中由存储器和密码选择逻辑产生的码。探针灯仅用于测试的目的，指示寄存器A和B、比较器的输出和寄存器C的输出的状态。

5. 以图9.42作为参考，判断图9.43中每个芯片的功能。

WWW 打开网络资源中系统应用文件示例的文件SAA09。使用Multisim软件运行安全密码逻辑仿真程序，观察结果。

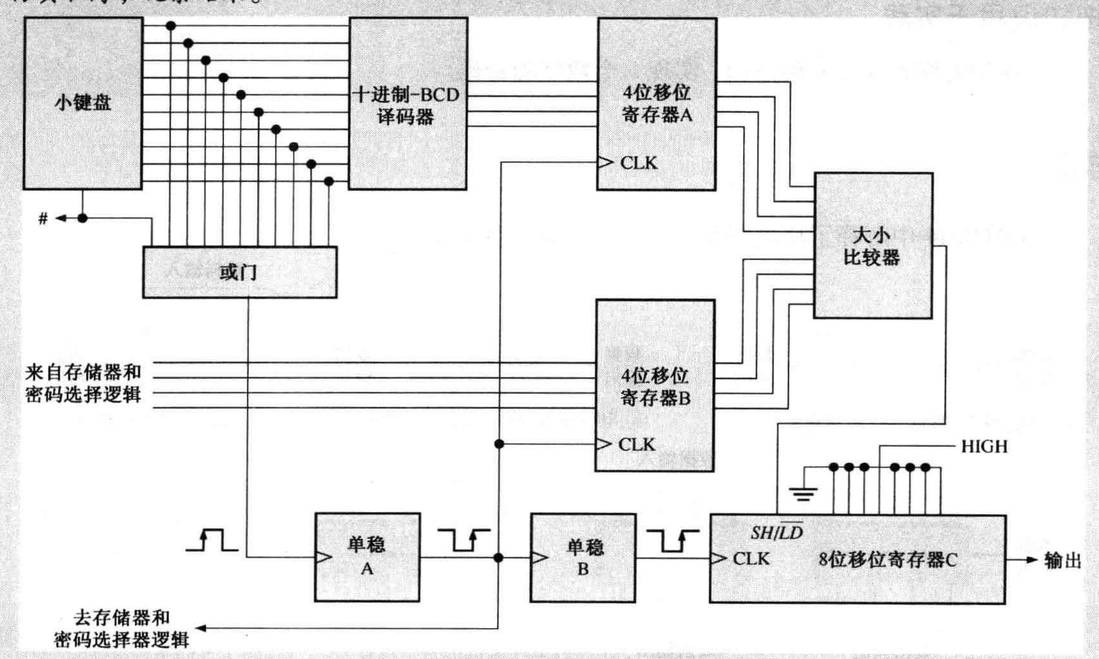


图9.42 安全密码逻辑的基本框图

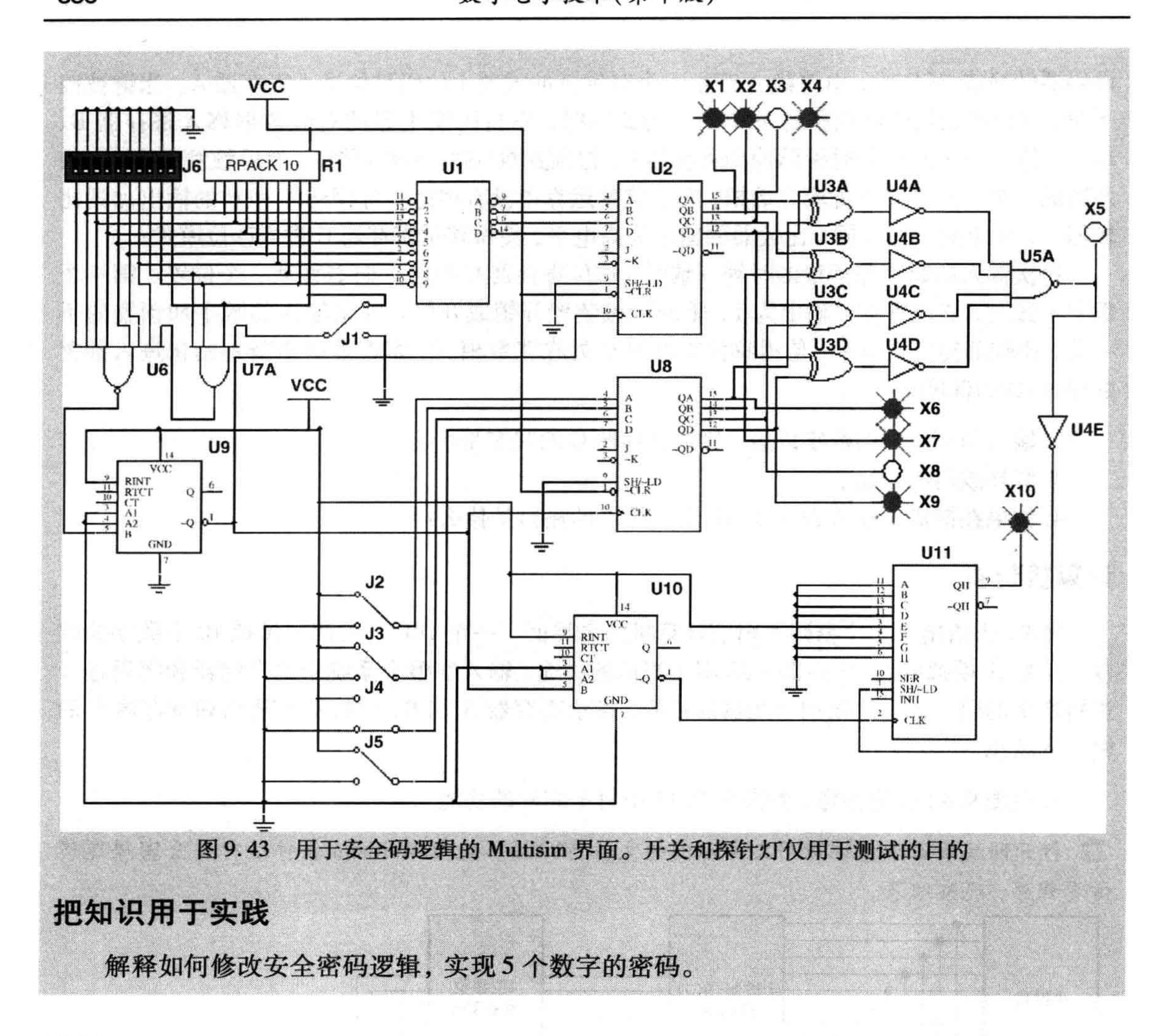


图 9.43 用于安全码逻辑的 Multisim 界面。开关和探针灯仅用于测试的目的

把知识用于实践

解释如何修改安全密码逻辑，实现 5 个数字的密码。

总结

- 在图 9.44 中解释了移位寄存器中数据移动的基本类型。

数据输入 → 数据输出
(a) 串行输入/移位 右/串行输出

数据输出 ← 数据输入
(b) 串行输入/移位 左/串行输出

数据输入 → 数据输出
(c) 并行输入/串行输出

数据输入 → 数据输出
(d) 串行输入/并行输出

数据输入 → 数据输出
(e) 并行输入/并行输出

(f) 右循环

(g) 左循环

图 9.44

- 移位计数器是具有特殊序列的反馈的移位寄存器，约翰逊计数器和环形计数器就是这样的例子。
- 约翰逊计数器的序列有 $2n$ 级，n 表示级的数目。
- 环形计数器在序列中有 n 级。

关键词

双向 有两个方向。在双向移位寄存器中，存储的数据可以右移也可以左移。

置数 把数据放入移位寄存器中。

寄存器 一个或多个触发器用于存储数据和移位数据。

级 寄存器中的一个存储单元。

判断题 （答案在本章的结尾。）

1. 移位寄存器由一组触发器组成。
2. 移位寄存器的两个功能是数据存储和数据移动。
3. 在串行移位寄存器中，几个数据位同时被输入。
4. 所有移位寄存器有特殊的序列定义。
5. 一个移位寄存器可以具有并行和串行输出。
6. 具有 4 级的一个移位寄存器可以存储最多 15 个数据。
7. 约翰逊计数器是一种特殊类型的移位寄存器。
8. 一个 8 位约翰逊计数器的模为 8。
9. 一个环形计数器为序列中的每一个状态使用一个触发器。
10. 一个移位寄存器可以作为一个时间延迟电路。

自测题 （答案在本章的结尾。）

1. 移位寄存器中的一级由下面的哪一项组成？
   (a)锁存器 (b)触发器 (c)存储字节 (d)4 位存储
2. 为了将 1 字节数据串行移位到移位寄存器中，必须有
   (a)1 个时钟脉冲 (b)1 个置数脉冲
   (c)8 个时钟脉冲 (d)数据中的每个 1 都要有 1 个时钟脉冲
3. 为了将 1 字节数据并行载入到一个具有同步载入的移位寄存器中，必须有
   (a)1 个时钟脉冲 (b)数据中的每个 1 都要有 1 个时钟脉冲
   (c)8 个时钟脉冲 (d)数据中的每个 0 都要有 1 个时钟脉冲
4. 一组数位 10110101 串行移位(首先移动最右边的位)到一个 8 位并行输出移位寄存器中，其初始状态为 11100100。在两个时钟脉冲之后，该寄存器的状态为
   (a)01011110 (b)10110101 (c)01111001 (d)00101101
5. 使用 100 kHz 的时钟频率，8 个数位可以在多少时间内串行进入移位寄存器中？
   (a)80 μs (b)8 μs (c)80 ms (d)10 μs
6. 利用 1 MHz 的时钟频率，8 个数位可以在多少时间内并行进入移位寄存器中？
   (a)在 8 μs 内 (b)在 8 个触发器的传输延迟时间内
   (c)在 1 μs 内 (d)在 1 个触发器的传输延迟时间内

7. 模 10 约翰逊计数器需要
   (a)10 个触发器　　(b)4 个触发器　　(c)5 个触发器　　(d)12 个触发器
8. 模 10 环形计数器至少需要
   (a)10 个触发器　　(b)5 个触发器　　(c)4 个触发器　　(d)12 个触发器
9. 当一个 8 位串行输入/串行输出移位寄存器用做 24 μs 的时间延迟时，时钟频率必须是
   (a)41.67 kHz　　(b)333 kHz　　(c)125 kHz　　(d)8 MHz
10. 图 9.38 中键盘译码电路的环形计数器的目的是
    (a)顺序为每一行加上高电平，以检测键是否按下
    (b)为键代码寄存器提供触发脉冲
    (c)顺序为每一行加上低电平，以检测键是否按下
    (d)顺序反偏压每一行中的二极管

习题

9.1 节　基本移位寄存器的功能

1. 为什么移位寄存器被认为是基本存储设备?
2. 可以保持两字节数据的寄存器的存储容量是多少?
3. 指出移位寄存器的两个功能。

9.2 节　串行输入/串行输出移位寄存器

4. 序列 1011 加到一个 4 位串行移位寄存器的输入，寄存器的初始状态为 0。一个时钟脉冲以后移位寄存器中的状态是什么?
5. 对于图 9.45 中的数据输入和时钟，确定图 9.3 中移位寄存器的每一个触发器的状态，并给出 Q 波形。假设该寄存器初始时状态为全 1。

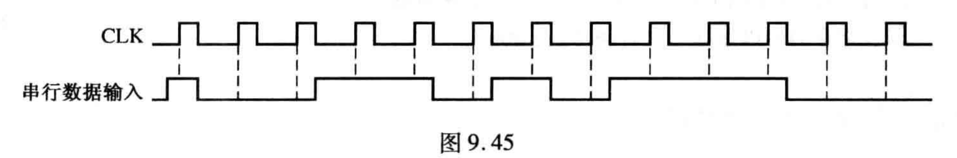


图 9.45

6. 对于图 9.46 中的波形，解出习题 5。

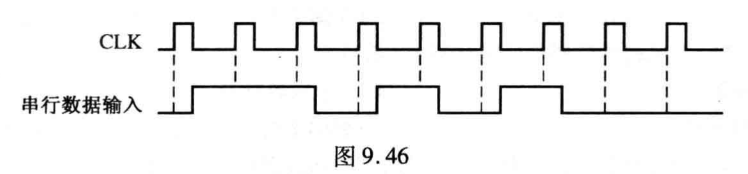


图 9.46

7. 如果图 9.47 中寄存器开始于 101001111000 状态，那么其在每个时钟脉冲之后的状态是什么?

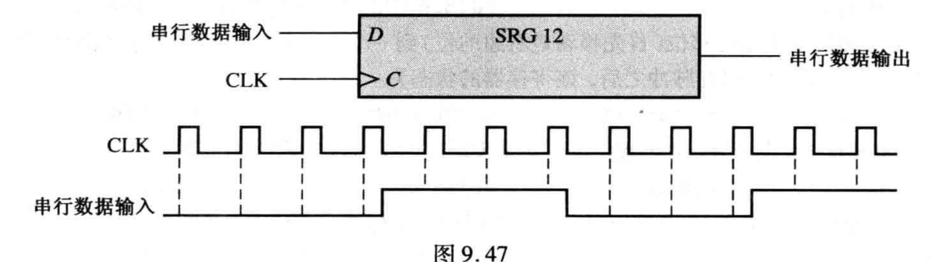


图 9.47

8. 对于串行输入/串行输出移位寄存器，确定图9.48中数据输入和时钟波形的数据输出波形。假设该寄存器初始时为清零状态。

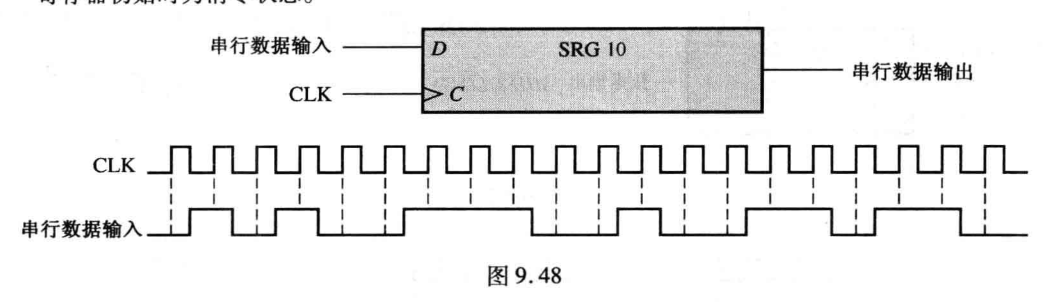


图9.48

9. 对于图9.49中的波形，解出习题8。

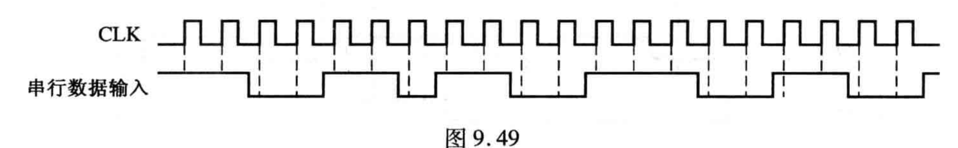


图9.49

10. 一个时钟上升沿触发的串行输入/串行输出移位寄存器具有如图9.50所示的数据输出波形。如果第一个数据位(最左面)是LSB，存储在8位寄存器中的二进制数是多少?

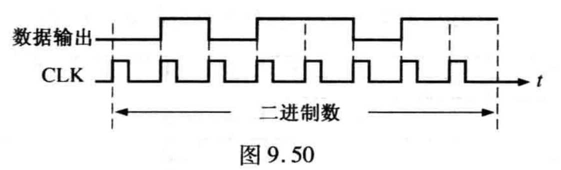


图9.50

9.3节　串行输入/并行输出移位寄存器

11. 给出一个完整的时序图，以表示图9.8中移位寄存器的并行输出。使用图9.48中的波形，寄存器初始时为清零。
12. 对于图9.49中的输入波形，解出习题11。
13. 为具有如图9.51所示输入波形的74HC164移位寄存器，绘制Q_0到Q_7的输出。

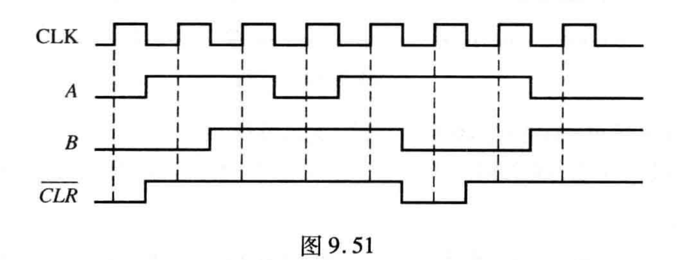


图9.51

9.4节　并行输入/串行输出移位寄存器

14. 图9.52(a)中的移位寄存器具有如图9.52(b)所示的移位/$\overline{置数}$($SHIFT/\overline{LOAD}$)和时钟输入。串行数据输入(SER)伪0。并行数据输入为$D_0=1$、$D_1=0$、$D_2=1$和$D_3=0$，如图所示。绘制出关于输入的数据输出波形。
15. 图9.53中的波形加在74HC165移位寄存器上。并行输入为全0，确定Q_7波形。
16. 如果并行输入为全1，解出习题15。
17. 如果SER输入反相，解出习题15。

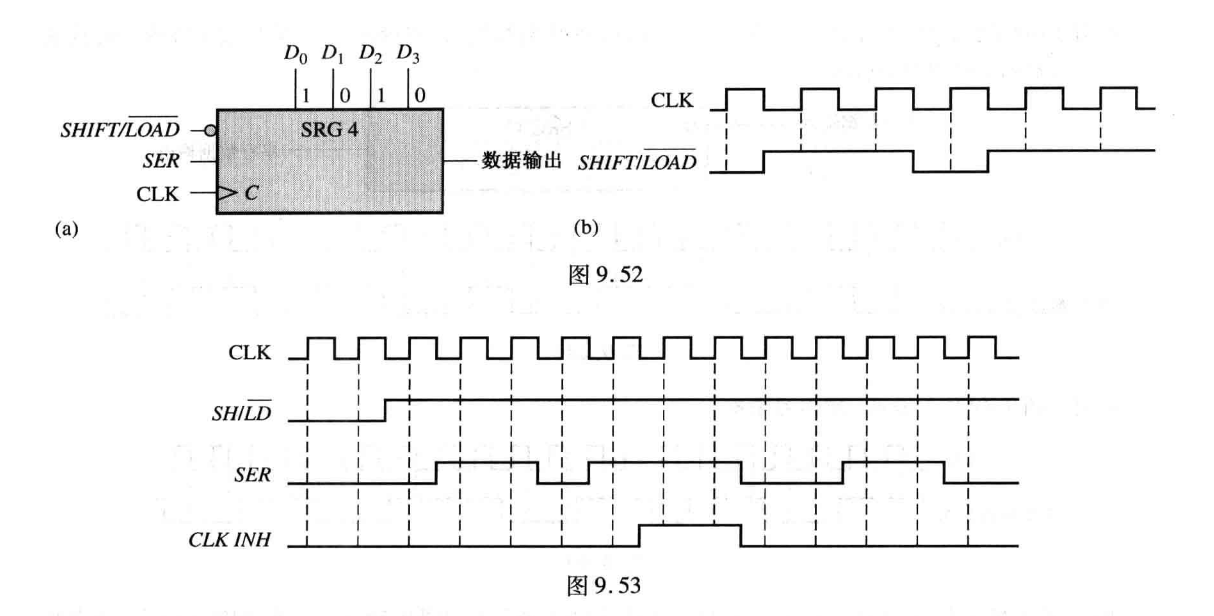


图 9.52

CLK
SH/LD
SER
CLK INH

图 9.53

9.5 节 并行输入/并行输出移位寄存器

18. 当输入如图 9.54 所示时，确定 74HC195 4 位移位寄存器的所有 Q 输出波形。

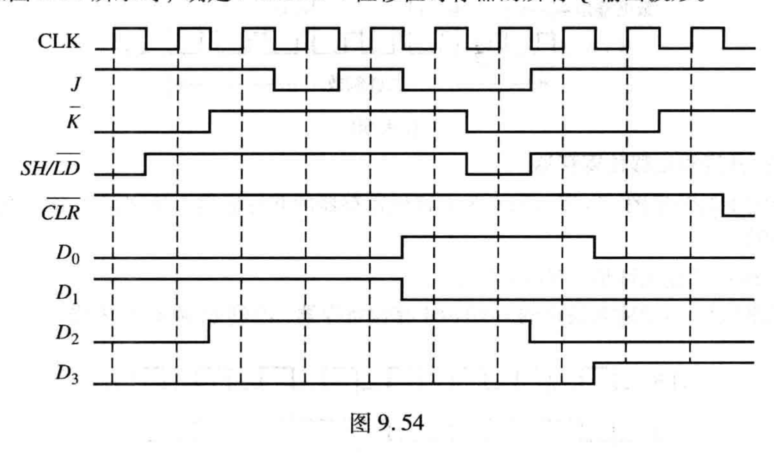


图 9.54

19. 如果 $SH/\overline{LD}$ 输入反相，而寄存器初始时清零，解出习题 18。
20. 使用两个 74HC195 移位寄存器来形成一个 8 位移位寄存器。给出所需要的连接。

9.6 节 双向移位寄存器

21. 对于图 9.55 中的 8 位双向寄存器，给定了 $RIGHT/\overline{LEFT}$ 控制波形，确定寄存器在每个时钟脉冲之后的状态。这个输入上的高电平启动向右移位，而低电平启动向左移位。假设该寄存器初始时存储了二进制形式的十进制数 76，并且最右边的位置为最低有效位(LSB)。数据输入线上有一个低电平。

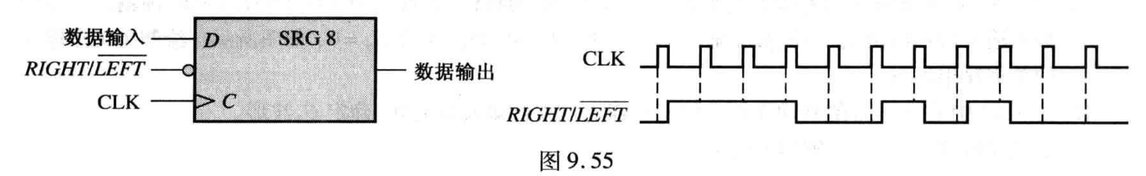


图 9.55

22. 对于图 9.56 中的波形，解出习题 21。

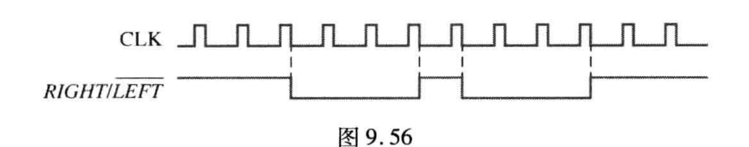


图9.56

23. 使用两个74HC194 4位双向移位寄存器来创建一个8位双向移位寄存器。给出连接结果。

24. 确定具有如图9.57所示输入的74HC194的Q输出。输入D_0、D_1、D_2和D_3都是高电平。

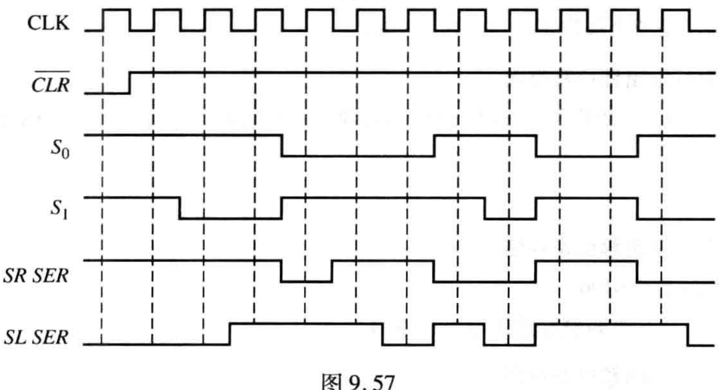


图9.57

9.7节　移位寄存器计数器

25. 实现下面每一个约翰逊计数器配置分别需要多少个触发器：

(a)模6　(b)模10　(c)模14　(d)模16

26. 绘制出模18约翰逊计数器的逻辑框图。给出时序图并以表格形式写出该时序。

27. 对于图9.58中的环形计数器，给出关于时钟的每个触发器的波形。假设FF0初始时为置位，其余的为复位。至少绘制10个时钟脉冲。

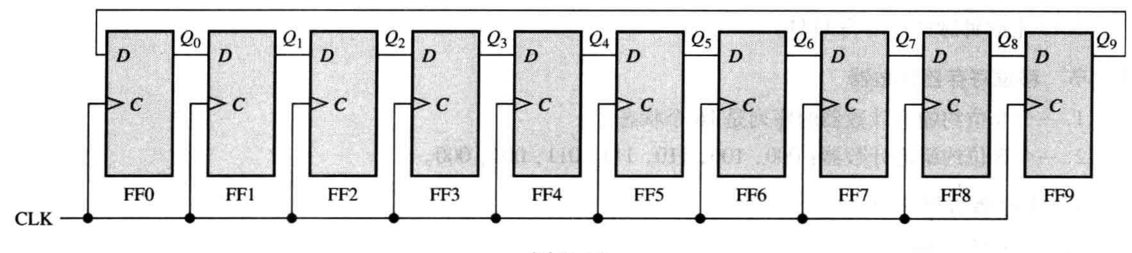


图9.58

28. 需要如图9.59所示的波形模式。设计一个环形计数器，并指出怎样进行预置才能在它的Q_9输出上产生这个波形。在CLK16处，该模式开始重复。

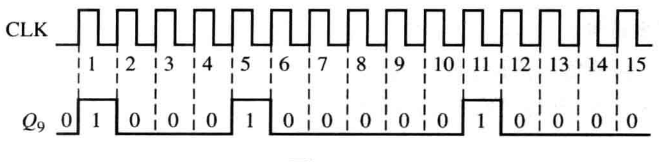


图9.59

9.8节　移位寄存器应用

29. 使用74HC195 4位移位寄存器来实现一个16位环形计数器。给出连接结果。

30. 图9.38中电源打开，$\overline{LOAD}$输入的目的是什么？

31. 当图9.38中的两个键被同时按下时，将会发生什么？

答案

温故而知新

9.1 节 基本移位寄存器的功能

1. 一个计数器具有一个特殊状态序列，但是移位寄存器没有。
2. 存储和数据移动是一个移位寄存器的两个功能。

9.2 节 串行输入/串行输出移位寄存器

1. FF0 数据输入到 J_0，($\overline{数据输入}$)到 K_0；FF1：Q_0 到 J_1，$\overline{Q_0}$ 到 K_1；FF2：Q_1 到 J_2，$\overline{Q_1}$ 到 K_2；FF3：Q_2 到 J_3，$\overline{Q_2}$ 到 K_3。
2. 8 个时钟脉冲。

9.3 节 串行输入/并行输出移位寄存器

1. 两个时钟脉冲以后为 0100。
2. 对于串行运算，最右边的触发器作为串行输出。

9.4 节 并行输入/串行输出移位寄存器

1. 高电平时，每个时钟脉冲数据右移一位。低电平时，并行输入端的数据进入寄存器。
2. 并行置数运算是异步的，所以不依赖于时钟。

9.5 节 并行输入/并行输出移位寄存器

1. 数据输出是 1001。
2. 一个时钟脉冲以后，$Q_0 = 1$。

9.6 节 双向移位寄存器

1. 15 个时钟脉冲以后为 1111。

9.7 节 移位寄存器计数器

1. 一个 8 位约翰逊计数器的序列是 16 个状态。
2. 一个 3 位约翰逊计数器：000，100，110，111，011，001，000。

9.8 节 移位寄存器应用

1. 625 次扫描/秒。
2. $Q_5Q_4Q_3Q_2Q_1Q_0 = 011011$。
3. 二极管为把行线下拉为低电平提供单向通路，阻止行线上的高电平与开关矩阵的连接。电阻把列线上拉为高电平。

9.9 节 关联标注的逻辑符号

1. 因为处在 0 状态，没有输入依赖于模式输入。
2. 是的，由 4D 标注指出，并行置数和时钟同步。

例题的相关问题

9.1 参见图 9.60。

9.2 3 个附加的时钟脉冲之后，寄存器的状态是 0000。

9.3 参见图 9.61。

9.4 参见图 9.62。

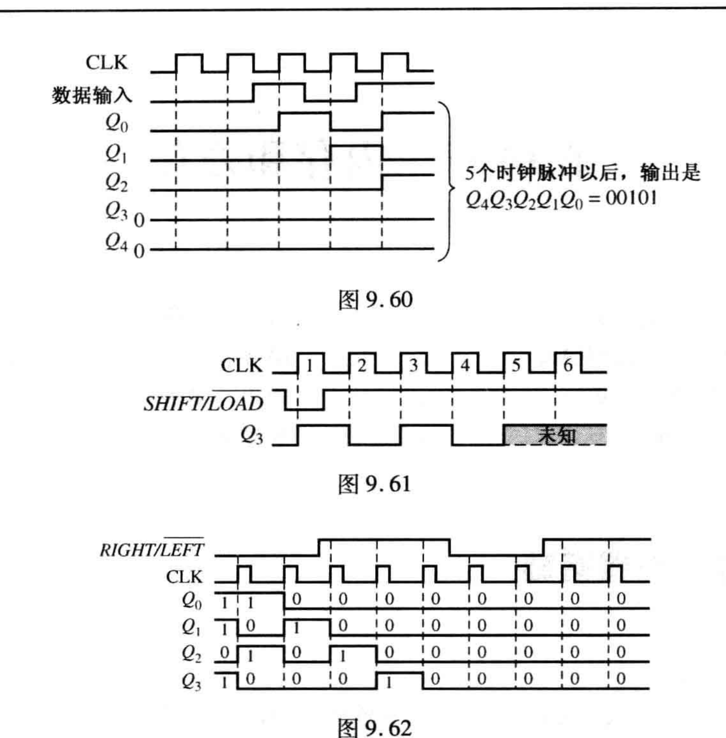


图 9.60

图 9.61

图 9.62

9.5　参见图 9.63。

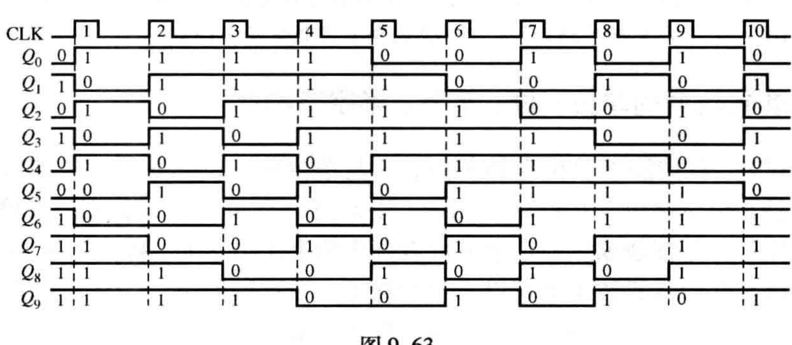


图 9.63

9.6　$f = 1/3\ \mu s = 333\ \text{kHz}$。

判断题

1. T　2. T　3. F　4. F　5. T　6. T　7. T　8. F　9. T　10. T

自测题

1. (b)　2. (c)　3. (a)　4. (c)　5. (a)
6. (d)　7. (c)　8. (a)　9. (b)　10. (c)

第 10 章　内存和外存

章节提纲

10.1　半导体存储器基础



计算机小知识

字的一般定义是：由二进制数据组成的一个完整的信息单位。当应用于计算机指令时，字就很明确地定义为两个字节(16 位)。作为计算机使用的汇编语言的重要组成部分，DW(定义字)指令的意思是以 16 位单元来定义数据。这个定义独立于特殊的微处理器或者数据总线的长。汇编语言还允许用 DB 指令来定义字节(8 位)，用 DD 指令来定义双字(32 位)，用 QD 指令来定义四字(64 位)。

10.1.1　二进制数据的单位：位、字节、半字节和字

作为一种规则，存储器以单位存储数据，这些单位从 1 位到 8 位。正如所知道的那样，二进制数据的最小单位是位。在许多应用中，以 8 位一个单元处理数据，该单元称为字节，或者以多个 8 位单元处理数据。一个字节可以分为两个 4 位单元，称为半字节。字节可以成组形成字。术语字在计算机技术中可以有两个意思。在存储器中，它定义为位的组合或者定义为可以在一个存储器中存储的以单个实体出现的字节。在汇编语言中，一个字被明确地定义为两个字节。

10.1.2　基本半导体存储阵列

存储器中的每一个存储元件都可以保存一个 1 或者 0，它们被称为一个单元。存储器由单元阵列组成，如图 10.1 所示，作为例子使用 64 个单元。存储阵列中的每个方块表示一个存储单元，并且它的位置通过指定行和列来确定。

64 单元阵列可以根据几种不同的基于数据单位的方式来组成。图 10.1(a)给出了一个 8×8 阵列，可以将其看成 64 位内存或者 8 字节内存。图 10.1(b)给出了一个 16×4 阵列，它是一个 16 个半字节内存，而图 10.1(c)给出了一个 64×1 阵列，它是一个 64 位内存。内存由其可以存储的字数乘以字长来指定。例如，16k×8 内存可以存储 16 384 个 8 位字。内存术语的矛盾在这里是很常见的。字的实际个数总是 2 的幂，也就是在这种情况下是 $2^{14}=16\ 384$。但是，通常的做法是以最接近千的数来表示这个数目，在这里是 16k。

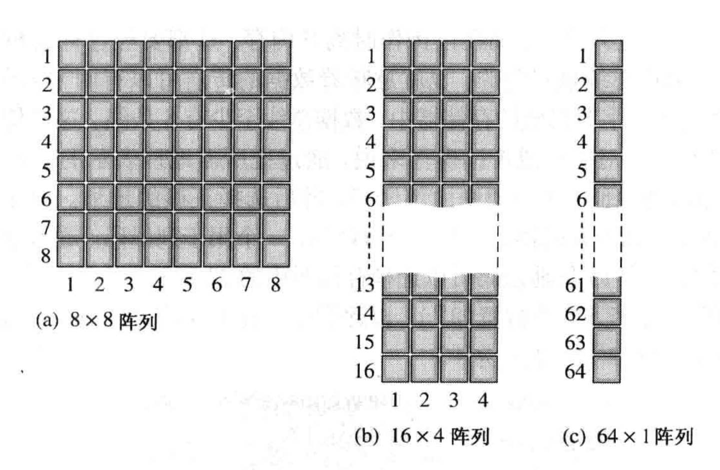


图 10.1　以 3 种不同方式组成的 64 单元内存

10.1.3　存储器地址和容量

存储阵列中数据单元的位置称为它的地址。例如，在图 10.2(a)中，阵列中某位的地址由如图所示的二维阵列的行和列来指定。在图 10.2(b)中，字节的地址只由行来指定。所以，正如所看到的那样，地址取决于存储器的数据单位是如何组成的。个人计算机具有以字节组成的随机存储器。这就意味着可以寻址的最小位组为 8 位。

在图 10.3 中，一个三维阵列的字节的地址如图所示，由行和列来指定。在此情况下，可以访问的最小位组为 8 位。

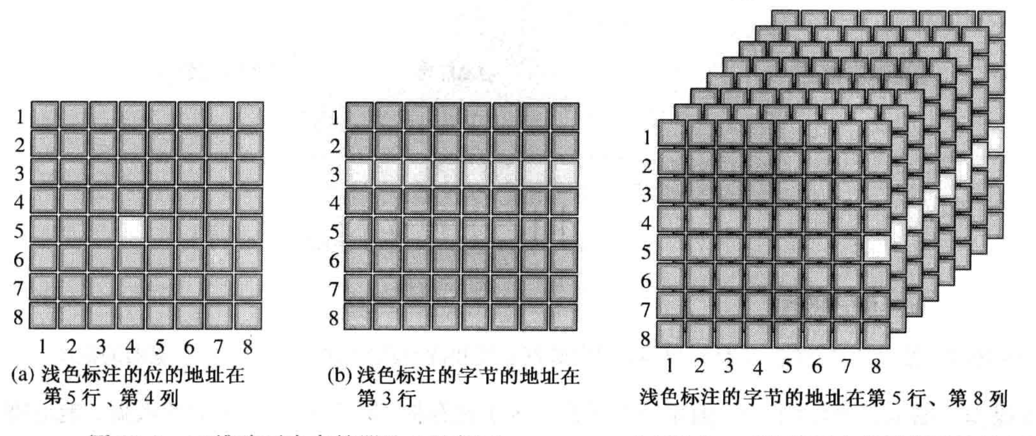


图 10.2　二维阵列中存储器地址的例子　　图 10.3　三维阵列中存储器地址的例子

存储器的容量是存储器可以存储的数据单位总数。例如，对于图 10.2(a)中以位组成的存储阵列，容量是 64 位。对于图 10.2(b)中以字节组织的存储阵列，容量是 8 字节，也是 64 位。在图 10.3 中，容量是 64 字节。计算机存储器一般具有 256 MB(兆字节)或者更多的内存。

10.1.4　存储器的基本操作

由于存储器存储二进制数据，所以必须把数据放到存储器中，并且在需要时从内存中复制数据。写操作将数据放到存储器中指定的地址，读操作把存储器中指定地址的数据复制出来。寻址操作是写操作和读操作的共同部分，用以选择指定的存储器地址。

数据单位在写操作时进入内存，而在读操作时离开内存，这都要经过称为数据总线的一系列线。如图 10.4 所示，数据总线是双向的，也就意味着数据的传送可以有两个方向(进入存储器或者离开存储器)。在这个以字节组成的存储器中，数据总线至少有 8 条线，这样使得选中地址中的 8 位数以并行的方式传送。对于写或者读操作来说，通过把所需地址表示的二进制码放到称为地址总线的一组线上来选择地址。地址码由内部译码，随后选择正确的地址。对于图 10.4(b)中的三维存储阵列的情况，有两个译码器，一个用于行译码，一个用于列译码。总线上线的数目由存储器的容量确定。例如，一个 15 位地址码可以选择存储器中的 32 768 个位置(2^{15})，一个 16 位的地址码可以选择存储器中的 65 536 个位置(2^{16})，以此类推。在个人计算机中，32 位的地址总线可以选择 4 294 967 296 个位置(2^{32})，表示为 4G。

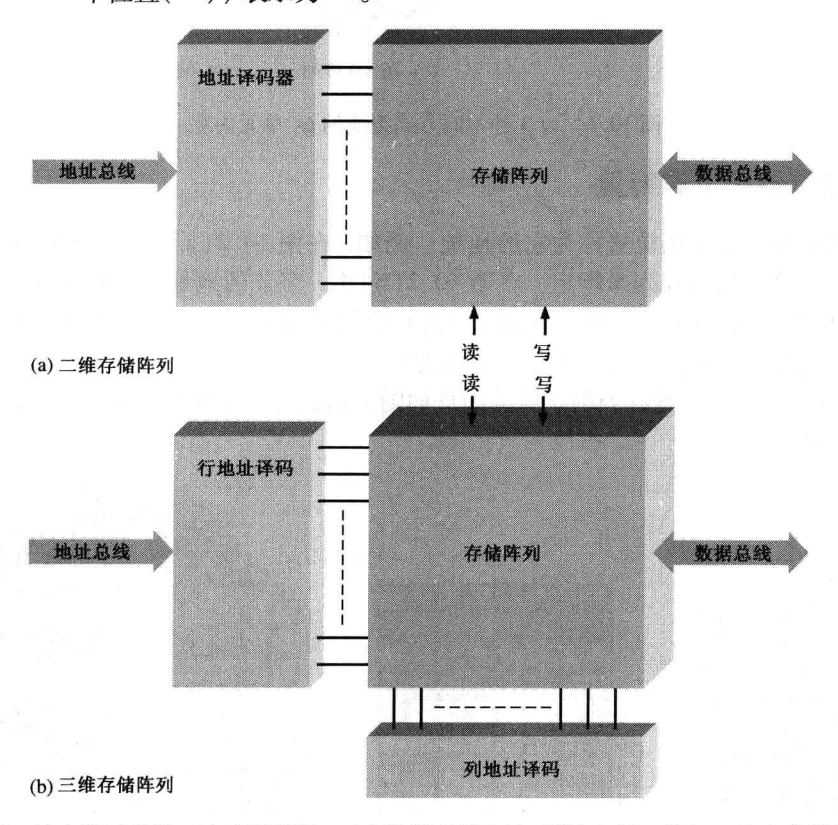


图 10.4 给出地址总线、地址译码器、双向数据总线和读/写输入的二维与三维存储器的框图

写操作 图 10.5 给出了一个简单的写操作。为了在存储器中保存一个字节的数据，地址寄存器所保存的代码被放到地址总线上。一旦地址码位于地址总线上，地址译码器就会对地址译码并且在内存中选择指定的位置。存储器随后得到一个写命令，数据寄存器中的数据字节就被放到数据总线上，并且存储在指定存储器地址中，因而完成了写操作。当新的数据字节写入存储器地址时，存储在那个地址上的当前数据字节就会被覆盖(由新的数据字节替代)。

读操作 一个简化的读操作如图 10.6 所示。同样，地址寄存器中的代码放到地址线上。一旦地址码位于总线上，地址译码器就会对地址译码并且选择存储器中指定的位置。存储器随后得到一个读命令，存储在所选择存储地址中的数据字节的一个"副体"被放到数据总线上，并放入数据寄存器，因此完成了读操作。当数据字节从存储器地址中读出时，它仍然保存在那个地址中。这称为非破坏性读出。

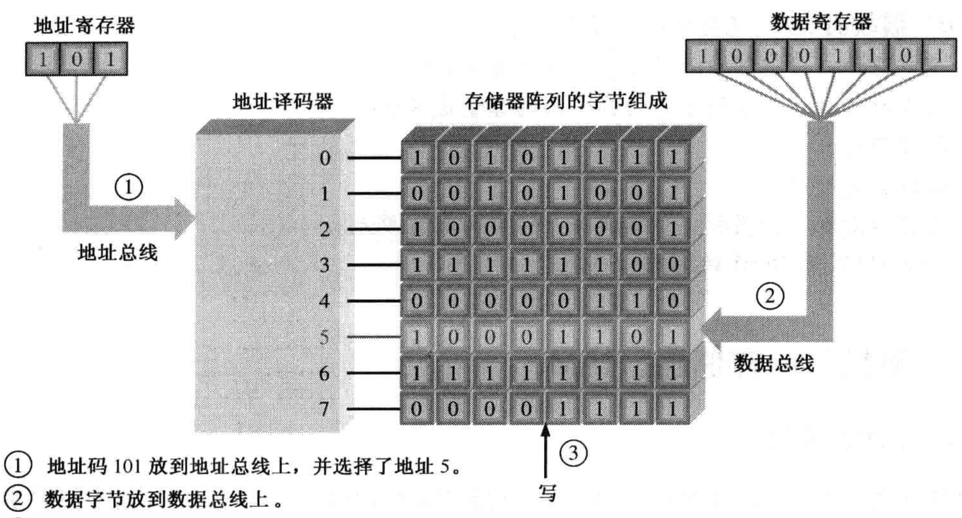


图 10.5　写操作的解释

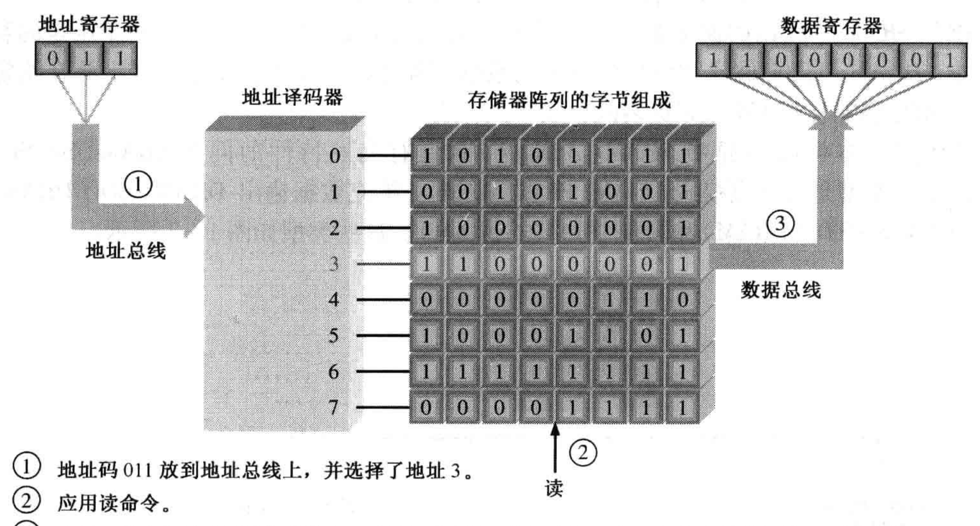


图 10.6　读操作的解释

10.1.5　RAM 和 ROM

两类主要的半导体存储器是 RAM 和 ROM。RAM(随机访问存储器)类型的存储器在等量时间内可以访问所有的地址，并且可以按照任何顺序来选择这些地址以执行读或者写操作。所有的 RAM 都具有读和写的能力。因为 RAM 在电源关闭时会丢失所存储的数据，所以它是易失性内存。

ROM(只读存储器)类型的存储器将永久或者半永久地保存数据。数据可以从 ROM 中读出，但是和 RAM 不同，它没有写操作。ROM 和 RAM 相似，也是随机存储器，但是术语 RAM 的传统意思是随机读/写存储器。本章将会介绍几种类型的 RAM 和 ROM。因为 ROM 在电源关闭后仍然保存着数据，所以它是永久性存储器。

10.1 节　温故而知新　(答案在本章的结尾。)

1. 可以存储在存储器中的最小数据单位是什么?
2. 可以存储256个数据字节的存储器的容量是多少?
3. 写操作是什么?
4. 读操作是什么?
5. 怎么得到在存储器指定位置的一个给定的数据单元?
6. 指出RAM和ROM的区别。

10.2　随机访问存储器(RAM)

10.2.1　RAM系列

RAM的两种类别是静态RAM(SRAM)和动态RAM(DRAM)。静态RAM通常使用锁存器作为存储元件,因而只要加上直流电源,就可以永远地存储数据。动态RAM使用电容作为存储元件,如果没有对电容再充电就不能长期存储数据,再充电则称为刷新过程。当直流电源移走后,SRAM和DRAM都会丢失存储的数据,因此被归类为易失性存储器。

数据从SRAM中读出的速度要比从DRAM中读出的速度快得多。但是,对于给定的物理空间和成本,DRAM可以存储比SRAM多得多的数据,因为DRAM单元更加简单,并在给定芯片面积里,因此可以比SRAM填充更多的单元。

SRAM的基本类型是异步SRAM(ASRAM)和具有突发特性的同步SRAM(SB SRAM)。DRAM的基本类型是快速页模式DRAM(FPM DRAM)、扩充数据输出DRAM(EDO DRAM)、突发EDO DRAM(BEDO DRAM)和同步DRAM(SDRAM)。这些类型如图10.7所示。

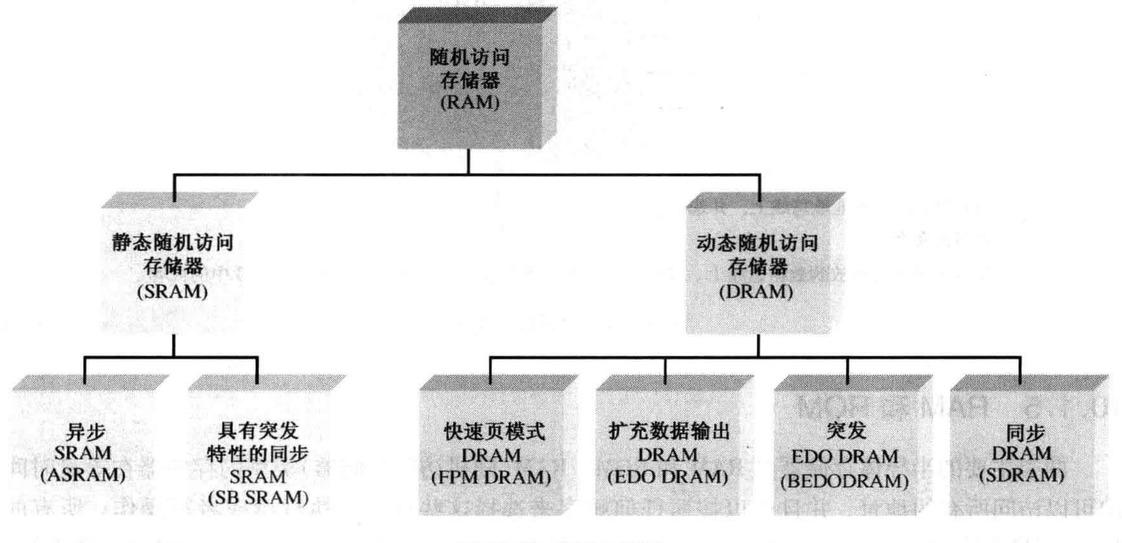


图10.7　RAM系列

10.2.2　静态RAM(SRAM)

存储单元　所有的静态RAM的特点是锁存器存储单元。只要直流电源加在SRAM上,它

就可以永远保持 1 或 0 的状态。如果电源移去，存储的数据位就会丢失。图 10.8 给出了一个基本的 SRAM 锁存器存储单元。这个单元由选择线上的有效电平所选择，数据位(1 或 0)通过数据写入线写入单元。一个数据位由数据读出线读出。

基本静态存储单元阵列　SRAM 中的存储单元由行和列组成，如图 10.9 所示为一个 $n\times4$ 阵列的情况。在每行中所有的单元享有相同的行选择线。在数据输出线中的每一组数据进入给定的行中的每个单元，并且连接到一条数据线，作为通过数据输入和数据输出缓冲器的一个数据输入和数据输出(数据 I/O)。

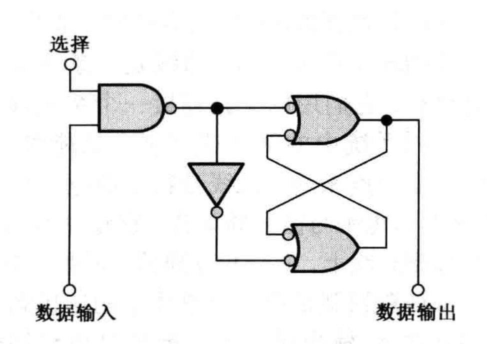


图 10.8　一个典型的 SRAM 锁存器存储单元

为了将一个数据单位(这里是半字节)写到内存阵列中给定的单元行中，行选择线就会进入它的有效状态，并且 4 个数据位被置于数据输入/输出(I/O)线上。这时写入线进入它的有效状态，使得每一个数据位保存到相关列的选中的单元上。为了读出一个数据单元，读出线进入它的有效状态，使得存储在选择行中的 4 个数据位出现在数据 I/O 线上。

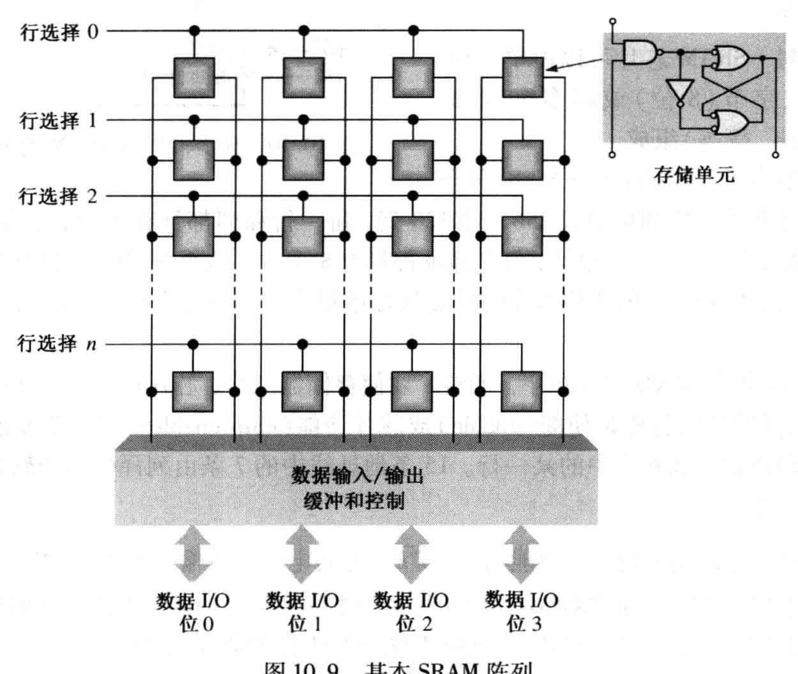


图 10.9　基本 SRAM 阵列

10.2.3　基本异步 SRAM 的组成

异步 SRAM 的操作并不同步于系统时钟。为了解释 SRAM 的一般组成结构，使用了一个 32k×8 位的内存。该内存的逻辑符号如图 10.10 所示。

在读模式时，存储在所选择地址中的 8 个数据位出现在数据输出线上。在写模式中，加在数据输入线上的 8 个数据位被存储到所选择的地址中。数据输入和数据输出线($I/O_0\sim I/O_7$)享用相同的线。在读期间，它们用做输出线($O_0\sim O_7$)；在写期间，它们用做输入线($I_0\sim I_7$)。

三态输出和总线 存储器中的三态缓冲器允许数据线用做输入或者输出线，并把存储器连接到计算机的数据总线上。这些缓冲器具有三个输出状态：高电平(1)、低电平(0)和高阻(开路)。三态输出在逻辑符号上由一个小的反三角(∇)来表示，如图 10.10 所示，它们用以和诸如在基于微处理器系统中的总线结构相兼容。

从物理学角度来看，总线是一条或多条传导路径，它们用以交互连接一个系统或者多个不同系统中的两个或者多个功能组件。从电子学角度来看，总线是特定的电压电平和/或当前电平与信号的集合，它允许不同设备连接到总线上，并且相互通信、正常工作。

微处理器就是由某个总线结构连接到存储器和输入/输出设备上。地址总线允许微处理器寻址存储器，而数据总线则提供了微处理器，存储器和输入/输出设备(如监视器、打印机、键盘、调制解调器)之间的数据传送。控制总线允许微处理器控制数据传送并且为不同的组件定时。

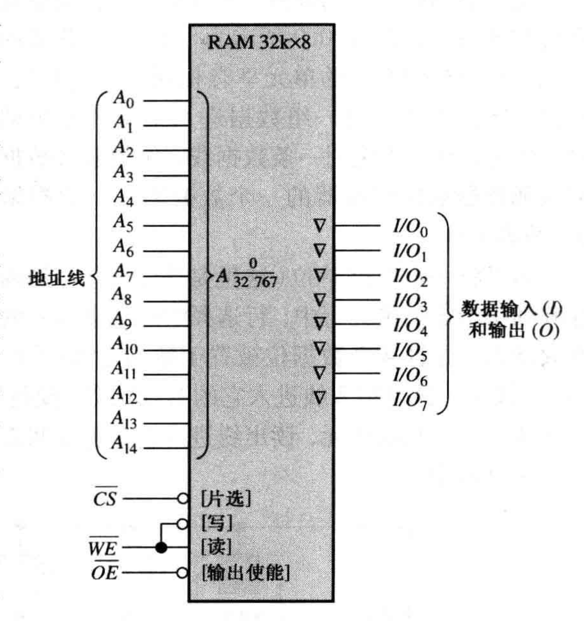


图 10.10 异步 32k×8 SRAM 的逻辑框图

存储器阵列 SRAM 芯片可以由单个位、半字节(4 位)、字节(8 位)或者多字节(16 位，24 位，32 位，等等)组成。

图 10.11 给出了典型的 32k×8 SRAM 的组成结构。存储器单元阵列排列成 256 行和128 列，每一行和列都具有 8 个位，如图 10.11(a)所示。实际上是 2^{15} = 32 768 个地址，每个地址都具有 8 个位。这个例子的存储容量是 32 768 个字节(一般表示为 32 kB)。虽然按当今的标准其容量很小，但是这个存储器的引入解释了基本的概念。

图 10.11(b)中的 SRAM 工作如下。首先，要使存储器工作，芯片选择(片选) $\overline{CS}$ 必须为低电平。[芯片选择的其他名称是使能(enable)或芯片使能(chip enable)。]15 条地址线中的 8 条由行译码器译码以选择 256 行中的某一行。15 条地址线中的 7 条由列译码器译码以选择 128 个 8 位列中的某一列。

读(READ) 在读模式时，写使能输入 ($\overline{WE}$) 为高电平，而输出使能 ($\overline{OE}$) 为低电平。输入三态缓冲器由门 G_1禁止，而列输出三态缓冲器由门 G_2开启。因此，来自所选地址的 8 个数据位途经列I/O到达数据线($I/O_0 \sim I/O_7$)，这时 $I/O_0 \sim I/O_7$作为数据输出线。

写(WRITE) 在写模式时，$\overline{WE}$为低电平而 $\overline{OE}$为高电平。输入缓冲器由门 G_1开启，而输出缓冲器由门 G_2禁止。因此，位于数据线上的 8 个输入数据位途经输入数据控制和列 I/O 到达所选择的地址中，并被保存。

读写周期 图 10.12 给出了存储器读周期和写周期的典型时序图。对于图 10.12(a)所示的读周期，一个有效地址码在一个特定的时间间隔加在地址线上，这个时间间隔称为读周期 t_{RC}。接下来，片选($\overline{CS}$)和输出使能($\overline{OE}$)输入变为低电平。在 $\overline{OE}$输入变为低电平后再经过一个时间间隔，来自所选择地址中的一个有效数据字节就会呈现在数据线上。这个时间间隔称为输出

使能存取时间 t_{GQ}。读周期的另外两种存取时间是地址存取时间 t_{AQ} 和芯片使能存取时间 t_{EQ}。t_{AQ} 从有效地址开始到数据线上出现有效数据为止；t_{EQ} 是从 $\overline{CS}$ 的高电平到低电平的转换到数据线上出现有效数据的时间。

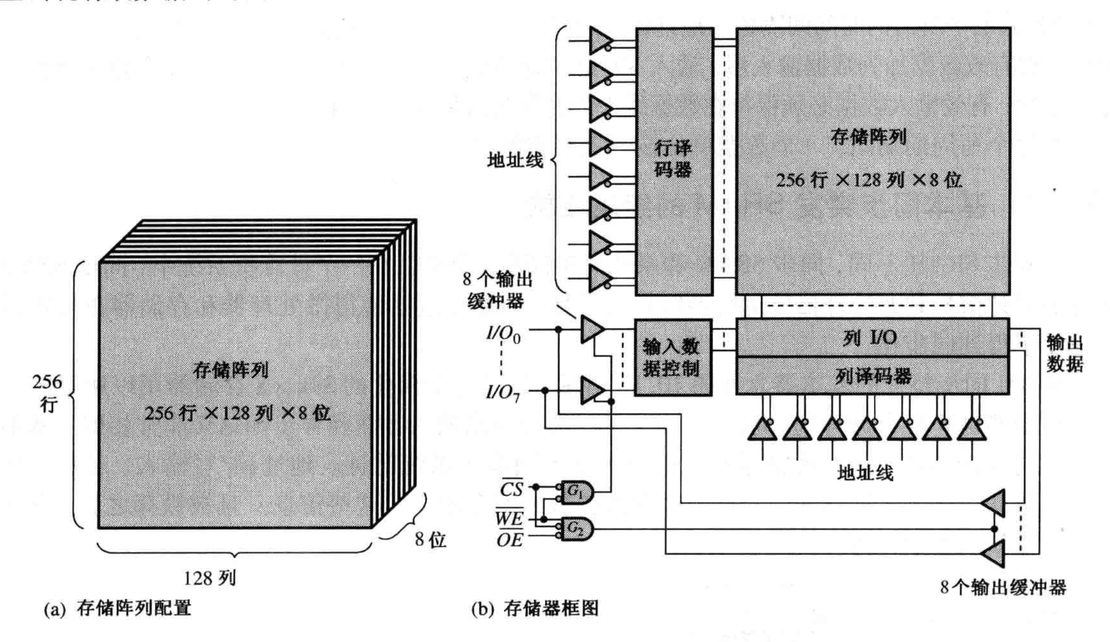


图 10.11　异步 32k×8 SRAM 的基本组成结构

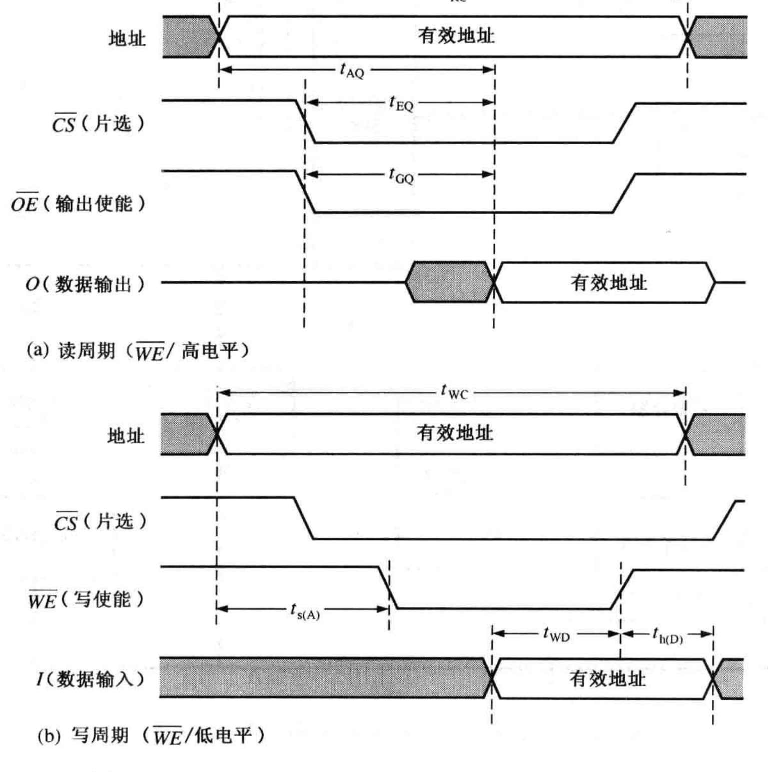


图 10.12　图 10.11 中 SRAM 的基本读写周期的时序图

在每个读周期期间，数据的一个单位(这里就是一个字节)就从存储器中读出。

对于图 10.12(b)的写周期，有效地址码在指定的时间间隔里加在地址线上，这个时间间隔称为写周期 t_{WC}。接下来，片选($\overline{CS}$)和写使能($\overline{WE}$)输入变为低电平。从有效地址开始直至输入 $\overline{WE}$ 变为低电平所需的时间间隔称为地址建立时间 $t_{s(A)}$。而 $\overline{WE}$ 输入必须为低电平的时间是写脉冲宽度。在有效数据加到数据输入后，输入 $\overline{WE}$ 必须保持低电平的时间命名为 t_{WD}；而在输入变为高电平之后，有效输入数据必须保持在数据线上的时间为数据保持时间 $t_{h(D)}$。

在每个写周期期间，一个数据单位被写入存储器中。

10.2.4 基本同步突发 SRAM 的组成结构

和异步 SRAM 不同，同步 SRAM 和系统时钟同步。例如，在一个计算机系统中，同步 SRAM 运行的时钟信号和微处理器所运行的时钟信号是一样的，这就使得微处理器和存储器在较快运行速度下得到同步化。

SRAM 同步特征的基本概念如图 10.13 所示，这是一个简化的 32k×8 存储器用以解释的框图。同步 SRAM 在存储阵列、地址译码器、读/写及使能输入方面和异步 SRAM 非常相似。基本区别是同步 SRAM 使用时钟寄存器使所有的输入同步于系统时钟。地址读/写输入、芯片使能和输入数据在有效时钟脉冲边沿都被锁存到各自的寄存器中。这些信息一旦被锁存之后，存储器的操作就同步于时钟。

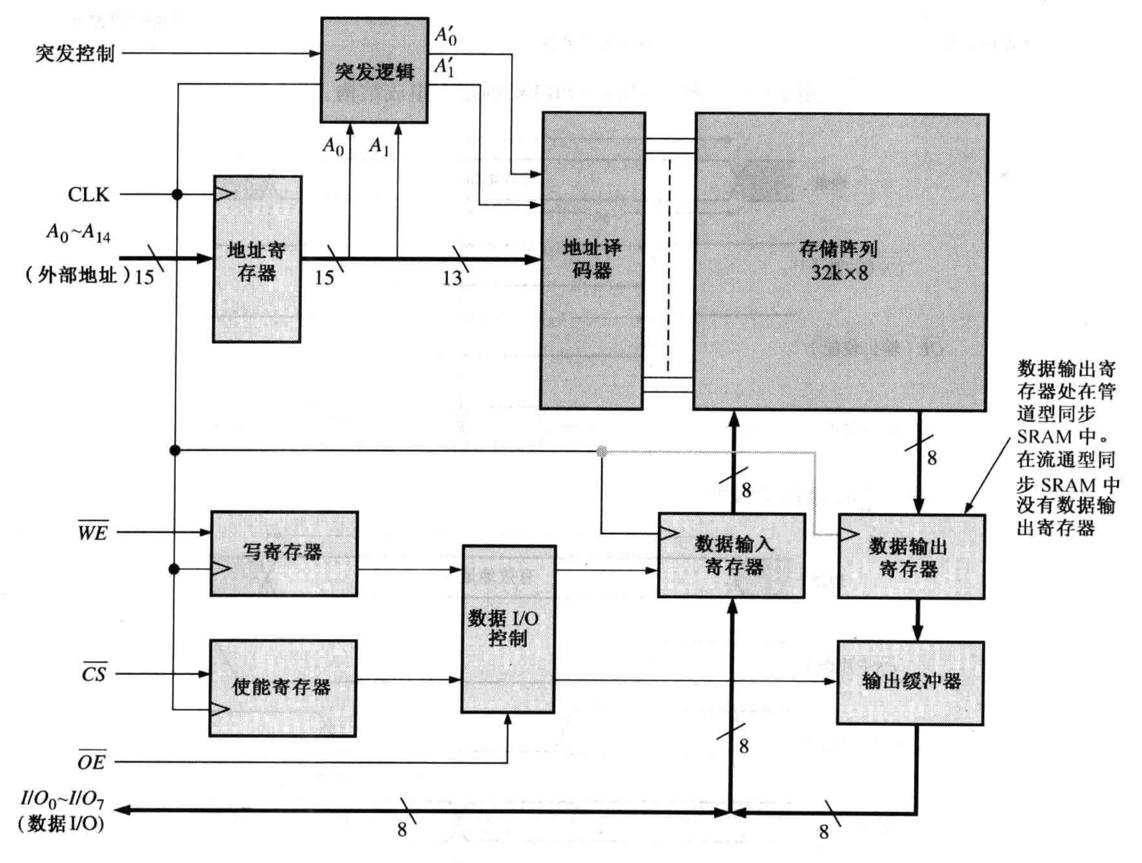


图 10.13 具有突发特性的同步 SRAM 的基本框图

出于简化的目的，图 10.13 引入了多并行线或者总线的标号，作为分别绘制每条线的替换方案。一组并行线可以由一条加粗线表示，加粗线上还带有一条斜线和独立线的个数。例如，下面的符号表示 8 条并行线的集合：

8

地址位 $A_0 \sim A_{14}$ 在时钟脉冲的上升沿锁存到地址寄存器中。在相同的时钟脉冲作用下，写使能($\overline{WE}$)线和片选($\overline{CS}$)的状态分别锁存到写寄存器和使能寄存器中。

这些都是 1 位寄存器或者简单的触发器。同样，在相同的时钟脉冲作用下，对于写操作，输入数据锁存到数据输入寄存器中；对于读操作，选中的存储器地址中的数据锁存到数据输出寄存器中，这都由基于输入的数据 I/O 控制来确定。这些输入来自写寄存器、使能寄存器和输出使能($\overline{OE}$)。

同步 SRAM 的两种基本类型是流通型(flow-through)和管道型。流通型(pipelined)同步 SRAM 并不具有数据输出寄存器，所以输出数据经过输出缓冲器异步流向数据 I/O 线。管道型同步 SRAM 具有数据输出寄存器，如图 10.13 所示，所以输出数据被同步置于数据 I/O 线上。

突发特性　如图 10.13 所示，同步 SRAM 一般具有地址突发特性，即允许存储器使用单个地址在 4 个序列位置进行读或者写。当外部地址锁存到地址寄存器时，两个最低位置的地址位 A_0 和 A_1 就加在突发逻辑上。在连续的时钟脉冲作用下，通过把 00、01、10 和 11 加到这两个最低顺序地址位上，产生 4 个内部地址的序列。此序列总是开始于基地址，也就是保留在地址寄存器上的外部地址。

典型的同步 SRAM 中的地址突发逻辑由一个二进制计数器和异或门组成，如图 10.14 所示。对于 2 位突发逻辑，由基地址位 $A_2 \sim A_{14}$ 再加两个突发地址位 A'_1 和 A'_0 就形成了内部突发地址序列。

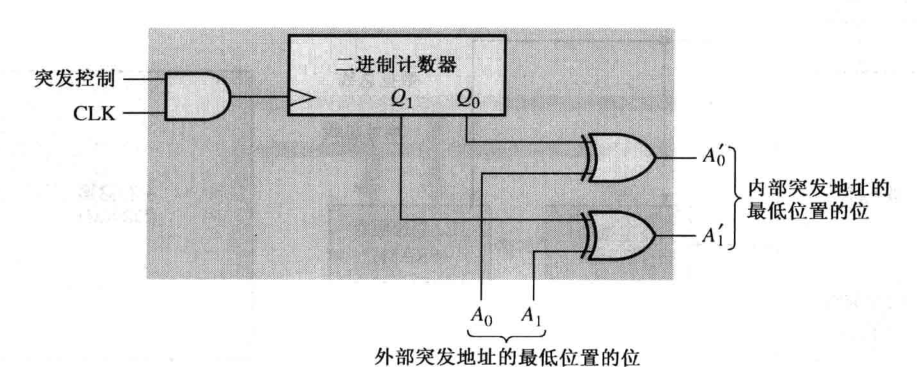


图 10.14　地址突发逻辑

为了开始这个突发序列，计数器处在它的 00 状态并且两个最低位置的地址位加在异或门的输入。假设 A_0 和 A_1 都是 0，两个最低位置的位表示的内部地址序列是 00、01、10 和 11。

10.2.5　高速缓冲存储器

SRAM 的一个主要应用是计算机中的高速缓冲存储器(cache memory)。高速缓冲存储器(简称高速缓存)是一种相对较小、高速的存储器，用来存储最近使用的指令或者来自较大而较慢的主存储器的数据。高速缓冲存储器还可以使用动态 RAM(DRAM)，随后将会介绍。典型情况下，SRAM 要比 DRAM 快好几倍。整体上，如果和只使用高容量 DRAM 的情况相比较，高速

缓冲存储器可以使得存储信息更快地进入微处理器。高速缓冲存储器基本上是改进系统性能的一个经济实惠的方法，而没有加快所有存储器速度所付出的代价。

高速缓冲存储器的概念基于这样一种想法：计算机程序倾向于在移向另一个区域之前，从主存储器中的某个区域取得指令或者数据。基本上，高速缓冲控制器“猜测”CPU(中央处理器)所需的下一个慢速动态存储器的区域，并将其移向高速缓冲存储器以备需要时使用。如果高速缓冲控制器的猜测正确，数据就可以立即用于微处理器。如果高速缓冲控制器猜测错误，CPU必须进入主存储器，为正确的指令或者数据等待更长的时间。幸运的是，高速缓冲控制器绝大多数情况下的猜测都是正确的。

高速缓存类比 描述高速缓存储可以使用许多类比，但是把它和家用电冰箱做比较可能是最有效的。家用电冰箱可以认为是某些食品项目的“高速缓存”，而超市就是主存储器，它是保存所有食品的地方。每次想吃或者喝什么东西时，就可以先看看冰箱(高速缓存)里有没有想要的东西。如果有，便节约了许多时间；如果没有，就不得不花费额外的时间去超市(主存储器)购买。

L1 和 L2 高速缓存 第一级高速缓存(L1 高速缓存)常常集成到处理器芯片上，并且具有非常有限的存储容量。L1 高速缓存也称为主高速缓存。第二级高速缓存(L2 高速缓存)是一个独立的外存储器芯片或者处理器之外的芯片集合，并且通常具有比 L1 高速缓存更大的存储容量。L2 高速缓存也称为次级高速缓存。某些系统可能具有更高级的高速缓存(L3，L4，等等)，但 L1 和 L2 是最常见的。同样，一些系统使用磁盘高速缓存来增强硬盘的性能，因为虽然 DRAM 比 SRAM 要慢得多，但是也要比硬盘驱动器快得多。图 10.15 阐释了计算机系统中的 L1 和 L2 高速缓存。

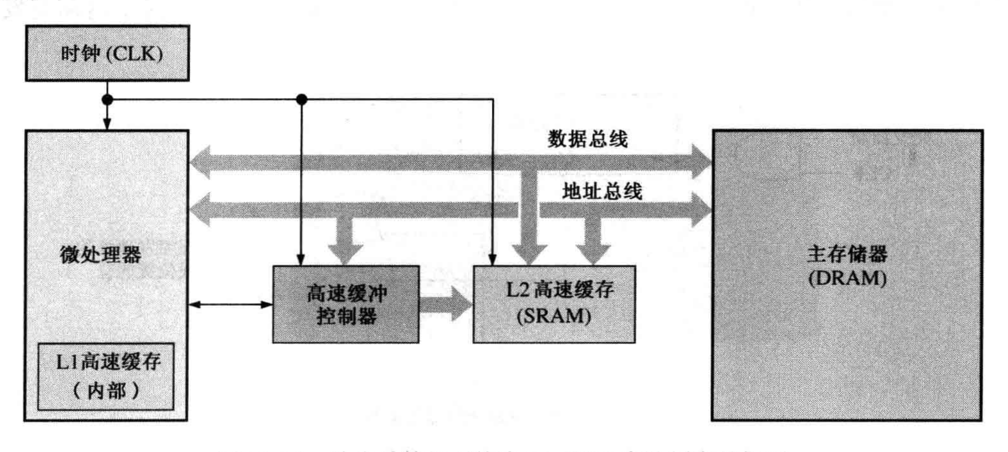


图 10.15 给出计算机系统中 L1 和 L2 高速缓存的框图

10.2.6 动态 RAM(DRAM)存储单元

动态存储单元在一个很小的电容里而不是锁存器里存储一个数据位。这种类型单元的优点是，它非常简单，因此允许很大的内存阵列构建在一个芯片上，并且每位的成本很低。缺点是存储电容器不能长时间保持它的电荷，并且会丢失它所保存的数据位，除非对它的电荷进行定期刷新。为了刷新就需要附加的存储电路，这使得 DRAM 的操作复杂化。图 10.16 给出了一个典型 DRAM 单元，由单个 MOS 晶体管(MOSFET)和一个电容组成。

在这种类型的单元中，晶体管作为开关来用。基本的简化操作如图 10.17 所示，并且如下所述。$R/\overline{W}$ 线(写模式)上的低电平使能三态输入缓冲器并且禁止输出缓冲器。对于即将写入单元中的一个 1 来说，D_{IN}(数据输入)线必须为高电平，必须由行线上的高电平使晶体管导通。晶体管相当于一个闭合开关，连接电容到位线上。这个连接允许电容充电到一个正电压，如图 10.17(a)所示。当 0 被存储时，低电平就加在 D_{IN}线上。如果电容器正在存储 0，它就保持非充电状态，如果它正在存储 1，电容就放电，如图 10.17(b)所示。当行线返回低电平时，晶体管截止，电容和位线断开，从而在电容器中“俘获”电荷(1 或者 0)。

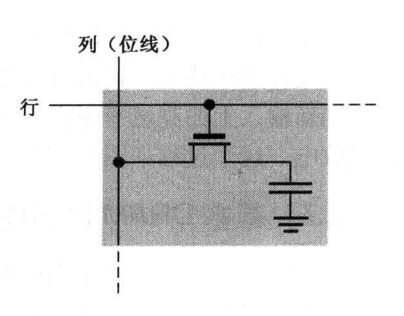


图 10.16　一个 MOS DRAM 单元

(a) 写一个 1 到存储单元

(b) 写一个 0 到存储单元

(c) 读一个 1 到存储单元

(d) 刷新一个存储的 1

图 10.17　一个 DRAM 单元的基本操作

为了从单元中读出，$R/\overline{W}$(读/写)线就是高电平，使能输出缓冲器并且禁止输入缓冲器。当行线取高电平时，晶体管导通并将电容器连接到位线上，因此也就连接到输出缓冲器(读放大器)上，所以数据位就出现在数据输出线(D_{OUT})上。这个过程如图 10.17(c)所示。

为了刷新存储单元，$R/\overline{W}$ 线是高电平，行线为高电平，而刷新线也是高电平。晶体管导通，把电容连接到位线上。输出缓冲器被开启，而存储的数据位加在刷新缓冲器的输入上，此缓冲器由刷新输入上的高电平开启。这就在相应存储位的位线上产生一个电压，因而重新对电容进行充放电。这个过程如图 10.17(d)所示。

10.2.7 基本 DRAM 的组成结构

DRAM 的主要应用是计算机的主存储器。DRAM 和 SRAM 之间的区别是存储单元的类型不同。正如所看到的那样，DRAM 的存储单元由一个晶体管和一个电容组成，这要比 SRAM 单元简单得多。这种组成结构允许 DRAM 中有更大的密度，因而在给定的芯片面积里有更大的位容量，但是存取时间慢了许多。

再一次，因为电容器中存储的电荷会泄漏，DRAM 单元需要经常地进行刷新操作来保持存储的数据位。这个需求就导致了比 SRAM 中更加复杂的电路。下面使用一般的 1M×1 位 DRAM 作为例子来讨论大多数 DRAM 常见的几个特征。

地址多路复用 DRAM 使用一种称为地址多路复用的技术来减少地址线的数目。图 10.18 给出了一个具有 1M×1 组成结构的 1 048 576 位(1 Mb)DRAM 的框图。这里着重于浅色的方块，以解释地址多路复用。深色的方块表示刷新逻辑。

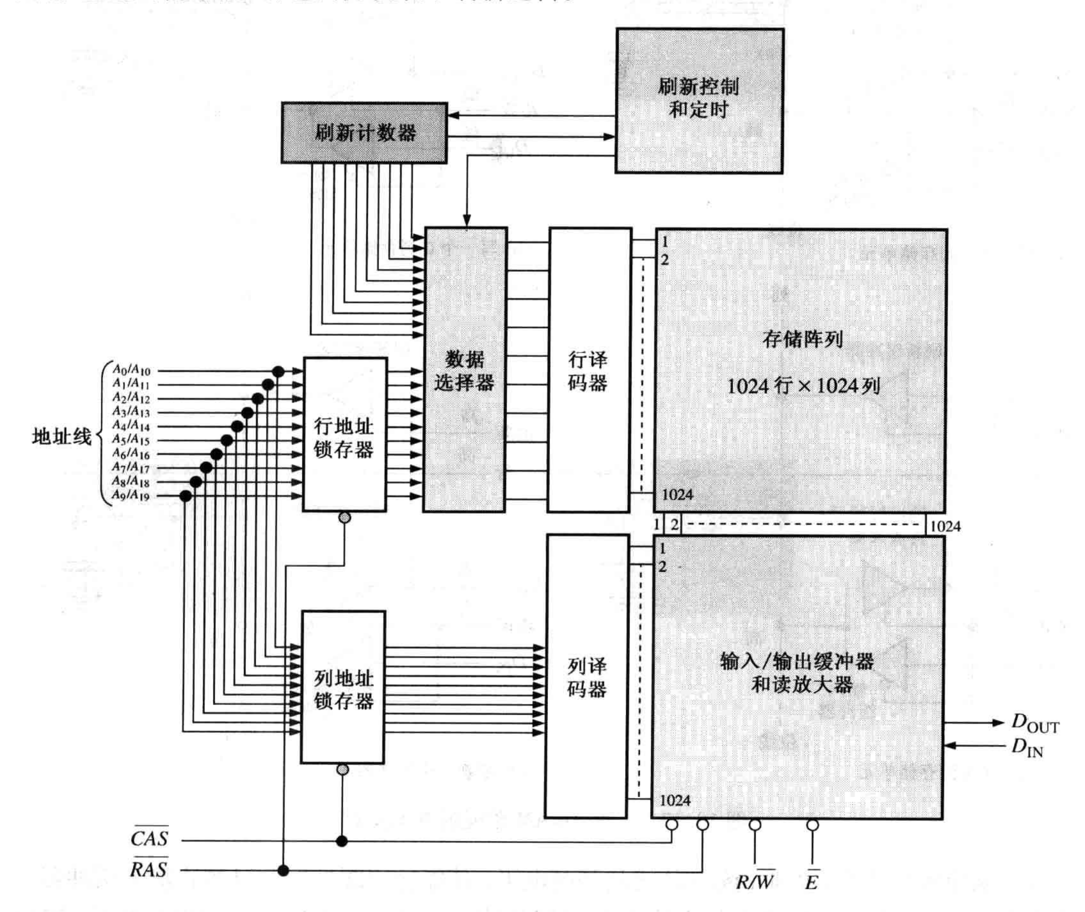


图 10.18 1M×1 DRAM 的简化框图

在存储周期的开始处，通过行地址选择（$\overline{RAS}$）和列地址选择（$\overline{CAS}$）而产生两个独立的 10 位地址字段，10 条地址线进行时分多路复用。首先，10 位行地址被锁存到行地址锁存器中。接下来，10 位列地址被锁存到列地址锁存器中。行地址和列地址被译码以后，在存储阵列中选择 1 048 576 个地址（2^{20} = 1 048 576）中的一个地址。该地址多路复用操作的基本时序如图 10.19 所示。

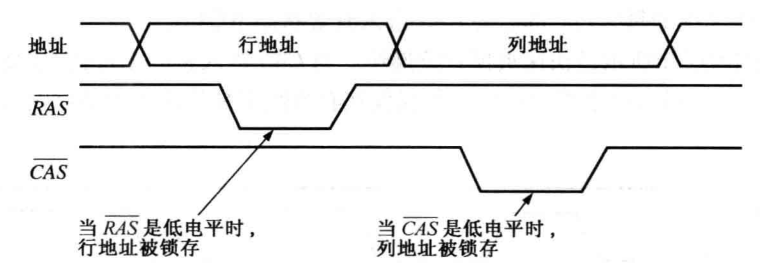


图 10.19　地址多路复用操作的基本时序

读和写周期　在每个读和写存储周期的开始，$\overline{RAS}$ 和 $\overline{CAS}$ 都进入有效状态（低电平）以多路复用行和列地址到锁存器和译码器。对于读周期，$R/\overline{W}$ 输入为高电平。对于写周期，$R/\overline{W}$ 输入为低电平。这个过程如图 10.20 所示。

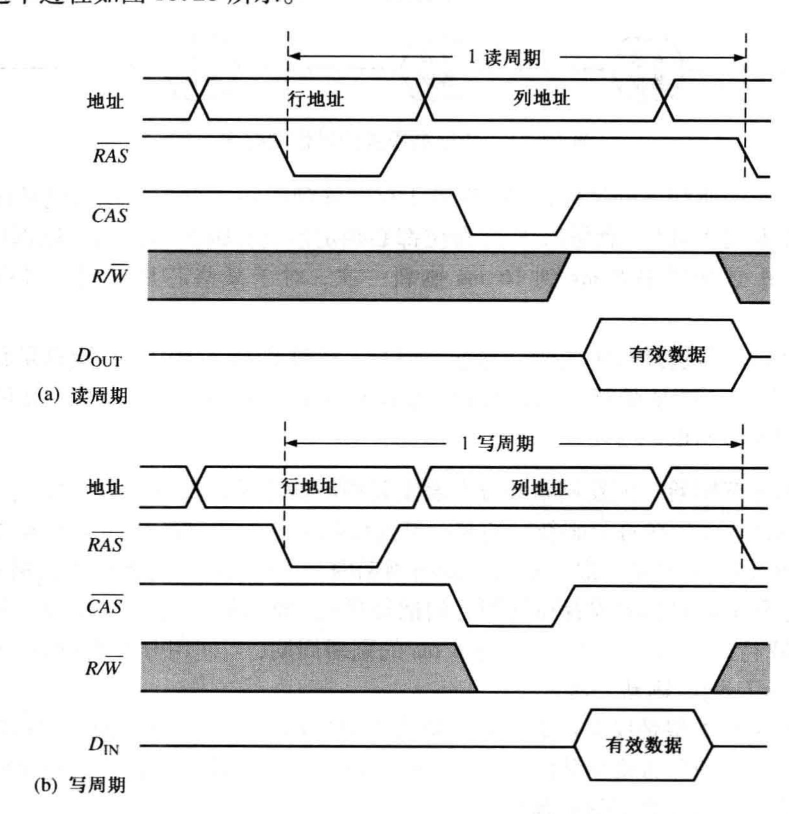


图 10.20　一般的读和写周期的时序

快速页模式　在先前所描述的一般读或者写周期中，特定存储位置的行地址首先由低电平有效 $\overline{RAS}$ 载入，随后此位置的列地址由低电平有效 $\overline{CAS}$ 载入。下一个位置由另外的 $\overline{RAS}$ 和 $\overline{CAS}$ 来选择，以此类推。

“页”就是在单个行地址中可用的存储部分，由一行中的所有列组成。快速页模式允许在所选择行中的每个列地址上进行快速连续的读或者写操作。通过 $\overline{RAS}$ 变为低电平并且在 $\overline{CAS}$ 在高低电平切换期间保持低电平，行地址首先被载入。单个行地址被选择，并且在 $\overline{RAS}$ 有效时保持被选中的状态。每个连续的 $\overline{CAS}$ 在所选择行中选择另外的列。所以，在一个快速页模式周期之后，所选择行中的所有地址都已经读出或者写入，这取决于 $R/\overline{W}$。例如，图 10.18 中DRAM 的快速页模式周期，对于 $\overline{RAS}$ 所选择的每个行，都需要 $\overline{CAS}$ 进入有效状态 1024 次。

如图 10.21 的时序图所示是快速页模式读操作。当 $\overline{CAS}$ 进入它的无效状态(高电平)时就禁止数据输出。因此，$\overline{CAS}$ 进入高电平的转换必须仅仅在有效数据被外部系统锁存以后才能发生。

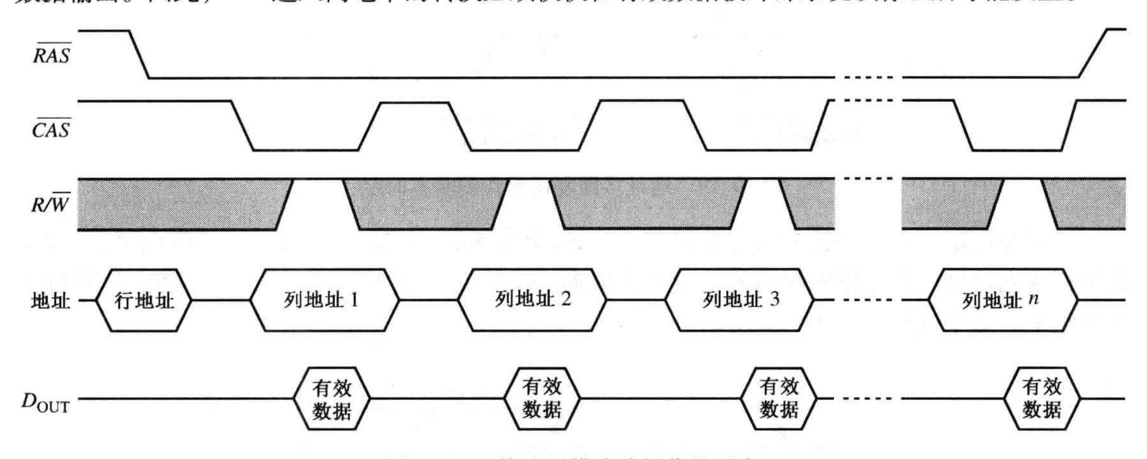


图 10.21　快速页模式读操作的时序

刷新周期　正如所知道的那样，DRAM 基于存储阵列中每一个位的电容电荷的存储。这些电荷随着时间和温度而减少(泄漏)，所以每位都必须定期进行刷新(再充电)以保持正确的位状态。通常，DRAM 必须每隔 8 ms 到 16 ms 刷新一次，对于某些芯片来说，刷新周期可以超过100 ms。

读操作自动刷新所选择行中的所有地址。但是，在典型的应用中，不能总是预期多久才会有一个读周期，所以不能依赖读周期足够频繁地出现来防止数据丢失。所以，必须在 DRAM 系统中实现特殊的刷新周期。

突发刷新和分布刷新　突发刷新和分布刷新是刷新操作的两种基本刷新模式。在突发刷新中，存储阵列中的所有行在每个刷新周期都会被连续刷新。对于刷新周期为 8 ms 的存储器，所有行的突发刷新每 8 ms 发生一次。在突发刷新周期内，一般的读和写操作都会被暂停。在分布刷新中，每一行都在正常的读或者写周期之间散布的时间间隔内被刷新。例如，图 10.18 中的存储器具有 1024 行。作为一个例子，对于 8 ms 的刷新周期，当使用分布刷新时，每一行都必须每隔 8 ms/1024 = 7.8 μs 刷新一次。

刷新操作的两种类型是仅 $\overline{RAS}$ 刷新和 $\overline{CAS}$ 先于 $\overline{RAS}$ 刷新。仅 $\overline{RAS}$ 刷新由 $\overline{RAS}$ 转换到低电平(有效)状态组成，其锁存将要刷新的行地址，而 $\overline{CAS}$ 在整个周期内保持为高电平(无效)。使用外部计数器提供这种操作类型的行地址。

$\overline{CAS}$ 在 $\overline{RAS}$ 刷新前，在 $\overline{RAS}$ 变为低电平前又通过 $\overline{CAS}$ 变为低电平，使得 $\overline{CAS}$ 初始化。这个序列使得内部刷新计数器工作，产生即将刷新的行地址。这个地址由数据选择器转换到行译码器中。

10.2.8　DRAM 的类型

现在，已经学习了 DRAM 的基本概念，下面简要地看看主要的类型。它们是快速页模式(FPM)DRAM、扩充数据输出(EDO)DRAM、突发扩充数据输出(BEDO)DRAM 和同步(S)DRAM。

FPM DRAM　快速页模式的操作先前已经描述过了。这种类型的 DRAM 传统上是最常见的，并且一直应用于计算机中，直至开发出了 EDO DRAM。回顾一下，存储器中的一页是指一个行地址内所包含的所有列地址。

FPM DRAM 的基本概念基于这样一种可能性：接下来要访问的几个内存地址位于同一行中(同一页中)。幸运的是，发生这样的情况所占的比例很高。FPM 比纯粹的随机存取要节省时间，因为在 FPM 中，行地址只被指定一次就可以访问几个连续的列地址，而对于纯粹的随机访问来说，对于每个列地址都要指定一个行地址。

记得在快速页模式读操作中，$\overline{CAS}$ 信号在可以进入它的无效状态之前不得不等待，直到来自给定地址的有效数据被外部系统(CPU)接收(锁存)为止。当$\overline{CAS}$进入它的无效状态后，数据输出就被禁止。这就意味着下一个列地址不能出现，直到来自当前列地址中的数据被传送到 CPU 之后为止。这就限制了页内的列寻址速度。

EDO DRAM　扩充数据输出 DRAM，有时也称为超页模式 DRAM，非常类似于 FPM DRAM。关键的区别是 EDO DRAM 中的 $\overline{CAS}$ 信号进入它的无效状态时，并不禁止输出数据，因为来自当前地址的有效数据可以保存下来，直至 $\overline{CAS}$ 再次有效。这就意味着在外部系统接收当前有效数据之前，可以访问下一个列地址。这种想法用以加速存取时间。

BEDO DRAM　突发扩充数据输出 DRAM 是一种具有地址突发能力的 EDO DRAM。从同步突发 SRAM 的讨论中，我们知道了地址突发特征，允许 4 个内部地址从单个外部地址中产生，这就节约了一些存取时间。这种相同的概念也适用于 BEDO DRAM。

SDRAM　为了跟上微处理器不断增加的速度，需要更快的 DRAM。同步 DRAM 就是实现这个目的的一种方式。和前面所讨论的同步静态 RAM 相似，SDRAM 的操作同步于系统时钟，这个时钟同时还驱动计算机系统中的微处理器。与同步突发 SRAM 有关的相同的基本思想也适用于 SDRAM。

这种同步的操作使得 SDRAM 完全不同于其他异步 DRAM 类型。使用异步存储，微处理器必须等待 DRAM 完成它的内部操作。但是，使用同步操作，DRAM 可以在系统时钟的控制下锁存地址、数据和来自处理器的控制信息。这就允许处理器在存储器的读或者写操作进行时处理其他的任务，而不是等待存储器去做它的事情，这一点和异步系统不同。

10.2 节　温故而知新

1. 列出 SRAM 的类型。
2. 高速缓冲存储器是什么？
3. 解释 SRAM 和 DRAM 的区别。
4. 描述 DRAM 中的刷新操作。
5. 列出 DRAM 的 4 种类型。

10.3 只读存储器(ROM)

10.3.1 ROM 系列

图 10.22 给出了半导体 ROM 是怎样归类的。掩模 ROM 在生产过程中将数据永久地存储在存储器中。PROM 也就是可编程 ROM，用户在专业化设备的帮助下，对数据进行电存储。掩模 ROM 和 PROM 可以使用 MOS 或者双极型技术。EPROM 也就是可擦除 PROM，是严格的 MOS 设备。紫外线(UV) EPROM 是用户电可编程的，但是存储的数据必须暴露在紫外灯下几分钟的时间，才可以被擦除掉。电可擦除 PROM(EEPROM 或者 E^2PROM)可以在几毫秒内将数据擦除掉。

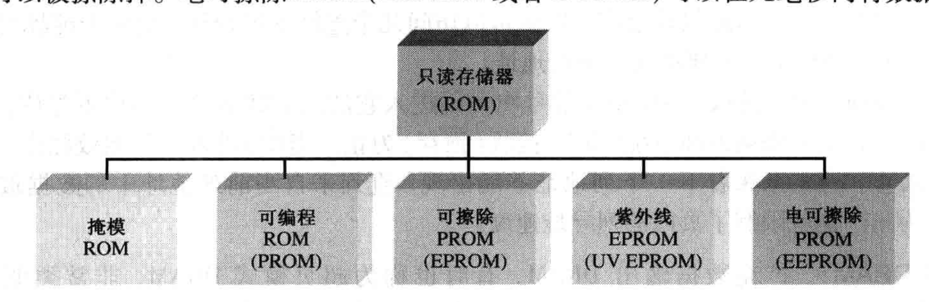


图 10.22 ROM 系列

10.3.2 掩模 ROM

掩模 ROM 通常简单地指 ROM。它在生产过程中被永久编程，以提供广泛使用的标准函数，比如通用变换，或者提供用户指定的函数。一旦存储器被编程之后，它就不能改变了。大多数 IC ROM 使用行/列联结处晶体管连接的存在或者缺失来表示 1 或者 0。

图 10.23 给出了 MOS ROM 单元。行线到晶体管栅极联结的存在表示该位置的一个 1，因为当行线为高电平时，所有栅极与这条行线连接的晶体管都导通，并将高电平连接到相关的列线上。在没有栅极连接的行/列联结处，当对行进行寻址时，列线保持为低电平(0)。

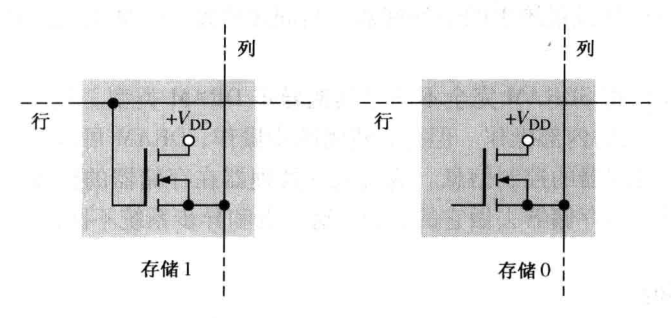


图 10.23 ROM 单元

10.3.3 简单的 ROM

为了解释 ROM 概念，图 10.24 给出了一个小型的简化 ROM 阵列。深色方格表示存储的是 1，而浅色方格表示存储的是 0。基本读操作如下所示：当二进制地址码加到地址输入线时，相应的行线就变为高电平。这个高电平在每个结点(单元)处通过晶体管连接到列线上，该单元存储的是 1。在每个存储 0 的单元处，由于终端电阻，列线保持为低电平。这些列线构成了数据输出。存储在所选行中的 8 个数据位出现在数据输出线上。

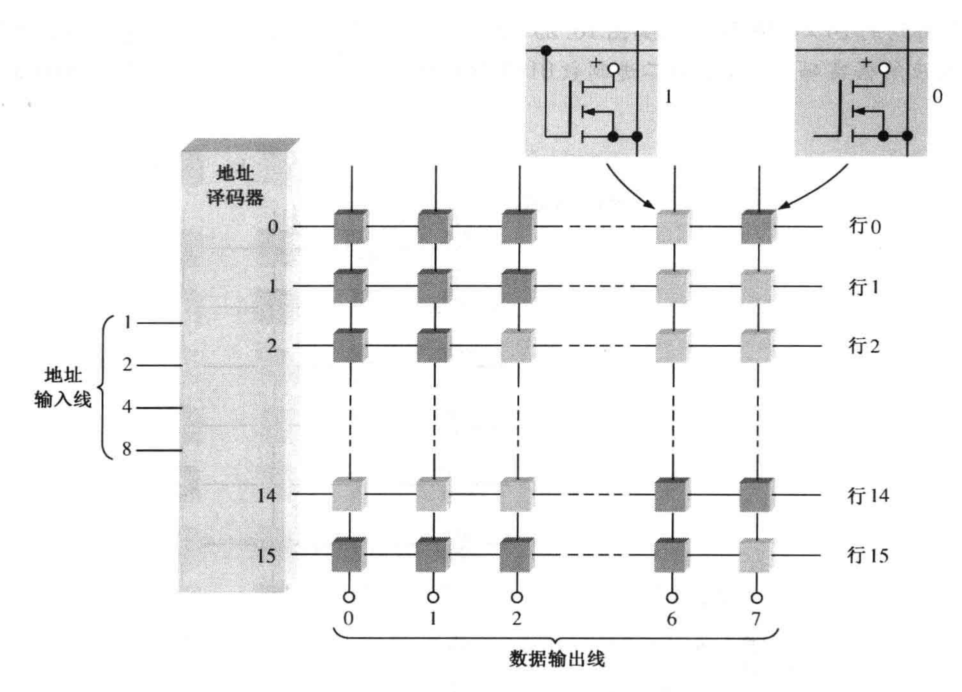


图 10.24　16×8 位 ROM 阵列的一个表示

正如所看到的那样，图 10.24 中的 ROM 被组成 16 个地址，每个地址保存 8 个数据位。因此，它是 16×8 ROM，并且它的总容量是 128 位或者 16 字节。ROM 可以用做查找表(LUT)，用于代码变换和逻辑函数的产生。

例 10.1　给出一个基本 ROM，和图 10.24 中的 ROM 相似，为 4 位二进制到格雷码的转换编程。

解： 复习第 2 章中的格雷码。开发出来的表 10.1 在可编程 ROM 中使用。

表 10.1

二进制				格雷码			
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

结果得到的 16×4 ROM 阵列如图 10.25 所示。可以看到地址输入线的二进制码产生输出线(列)上相应的格雷码。例如，当二进制数 0110 加在地址输入线上时，存储格雷码 0101 的地址 6 被选中。

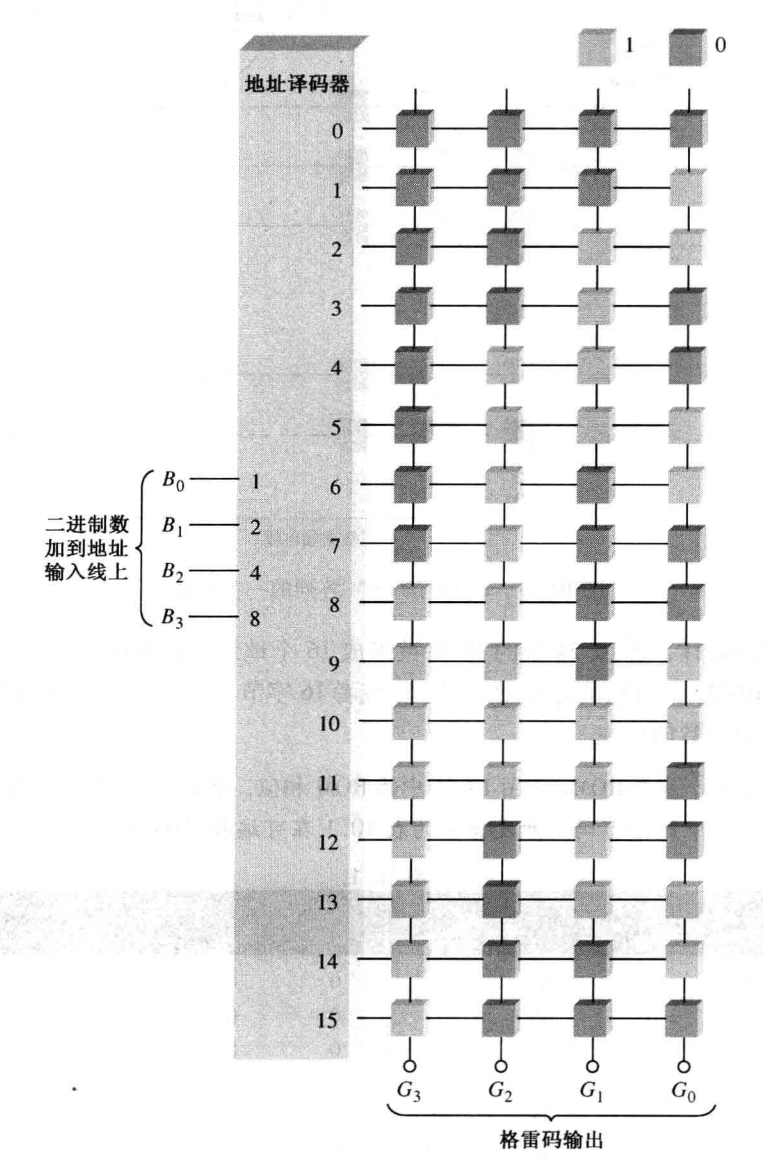


图 10.25 作为二进制到格雷码转换器的一个 ROM 编程的表示

相关问题①：利用图 10.25，当二进制数 1011 加到地址输入线上时，确定格雷码的输出。

10.3.4 内部 ROM 码组成结构

大多数 IC ROM 要比刚刚呈现的基本简化例子具有更加复杂的内部组成结构。为了解释 IC ROM 是怎样构建的，使用一个具有 256×4 组成结构的 1024 位芯片。逻辑符号如图 10.26 所

① 答案在本章的结尾。

示。当 256 个二进制代码(8 位)中的任何一个加到地址线上时，如果芯片使能输入为低电平，4 个数据位就会出现在输出上(因为 256 个地址需要 8 条地址线)。

虽然芯片的 256 ×4 组成结构说明在存储阵列中具有 256 行和 4 列，然而实际上并不是这样的。存储单元阵列实际是一个 32 ×32 的矩阵(32 行和 32 列)，如图 10.27 中的框图所示。

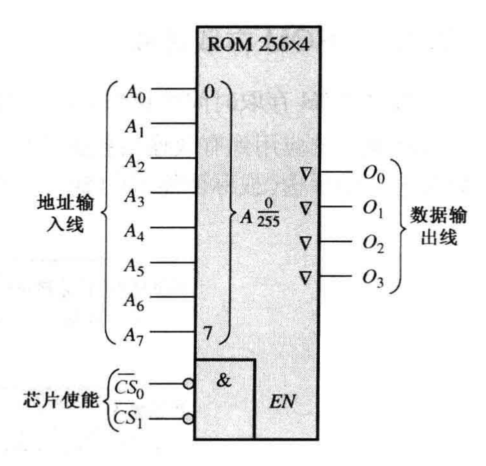


图 10.26　一个 256 ×4 ROM 的逻辑符号。指示符$A\frac{0}{255}$的意思是8位地址代码选择0 ~255的地址

图 10.27 中的 ROM 的工作如下：8 条地址线中的 5 条($A_0 \sim A_4$)由列译码器(经常称为 Y 译码器)来译码，以选择 32 列中的 1 列。8 条地址线中的 3 条($A_5 \sim A_7$)由列译码器(经常称为 X 译码器)来译码，以选择 32 列中的 4 列。实际上，列译码器由四个 8 选 1 译码器(数据选择器)组成，如图 10.27 所示。

这个结构的应用结果是，当应用一个 8 位地址代码($A_0 \sim A_7$)时，一个 4 位数据字就出现在数据输出上，这时芯片使能线($\overline{CS_0}$ 和 $\overline{CS_1}$)为低电平，以启动输出缓冲器。这种类型的内部组成结构(体系结构)是典型的不同容量的 IC ROM。

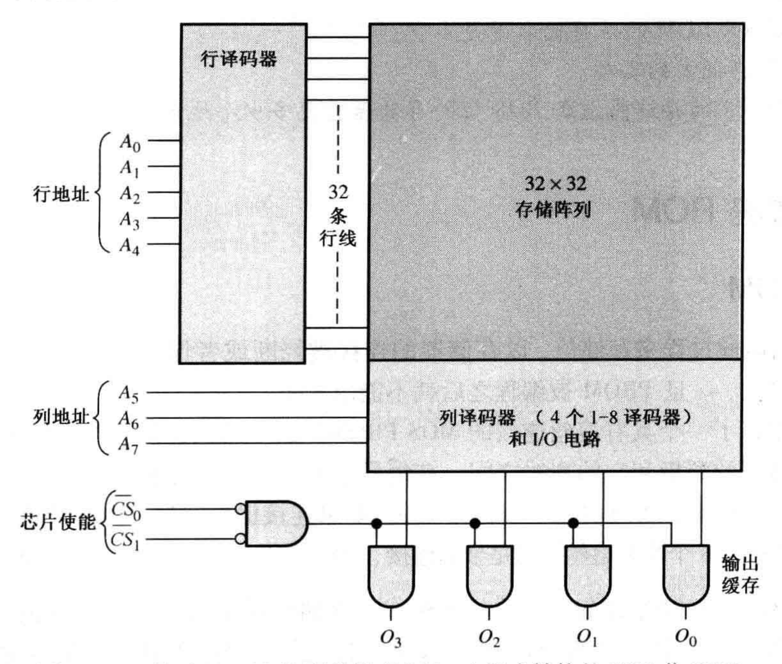


图 10.27　基于 32 ×32 阵列的具有 256 ×4 组成结构的 1024 位 ROM



计算机小知识

ROM 被用来存储 BIOS(基本输入/输出系统)。它们是一些程序，用以为计算机执行基本的监管和支持功能。例如，存储在 ROM 中的 BIOS 程序控制某种电视监视器的功能，提供磁盘格式化，为输入扫描键盘，以及控制某种打印机的功能。

10.3.5　ROM 存取时间

解释 ROM 存取时间的典型时序图如图 10.28 所示。ROM 的存取时间是 t_a，即从输入线上有效地址码的应用到有效输出数据的出现所需的时间。当有效地址已经处于输入线上时，也可以从芯片的片选(或称使能 $\overline{CS}$)输入到有效输出数据的出现这段时间来测量片选时间。

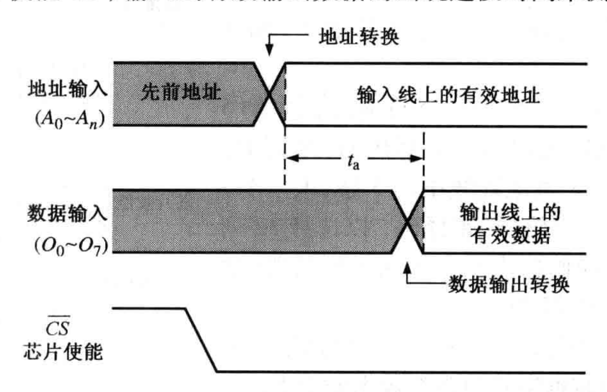


图 10.28　芯片使能有效，从地址改变到数据输出的 ROM 存取时间(t_a)

10.3 节　温故而知新

1. 一个 512 × 8 ROM 的位存储容量是多少?
2. 列出只读存储器的类型。
3. 一个 256 × 8 的存储器组成 2048 位的存储器需要多少个地址位?

10.4　可编程 ROM

10.4.1　PROM

PROM 利用熔丝过程来存储位，以存储器的连接被烧断或者保持完好来表示 0 或者 1。熔丝过程是不可逆的，一旦 PROM 被编程之后就不能改变了。

图 10.29 解释了一个具有熔丝连接的 MOS PROM。这个熔丝连接被安装到 PROM 中，位于每个单元的晶体管的源极和它的列线之间。在编程过程中，一个足够大的电流注入，通过熔丝的可熔断连接将其烧断，从而创建一个存储的 0。如果连接保持完好，就是存储 1。

用于 PROM 中的 3 个基本熔丝技术是金属连接、硅连接和 pn 结。每种技术的简短描述如下。

1. 金属连接由诸如镍铬合金之类的材料构成。存储阵列中的每个位都由独立的连接表示。在编程期间，连接要么“吹开式”断开要么保持完好。基本上首先对给定的单元寻址，然后迫使足够量的电流经过连接造成它断开。
2. 硅连接由狭窄、有凹槽的多晶硅条带组成。这些熔丝的编程需要将足够的电流通过它们，才能熔断连接。这些电流在熔丝部位产生高温，从而使硅氧化并在刚才断开的连接周围形成绝缘层。
3. 短路结或者雪崩诱导迁移技术，基本上由两个背靠背排列的 pn 结组成。在编程期间，二极管的一个结崩塌，结果电压和热量引起铝离子迁移和结的短路。其余的结随后用做正偏压二极管以表示一个数据位。

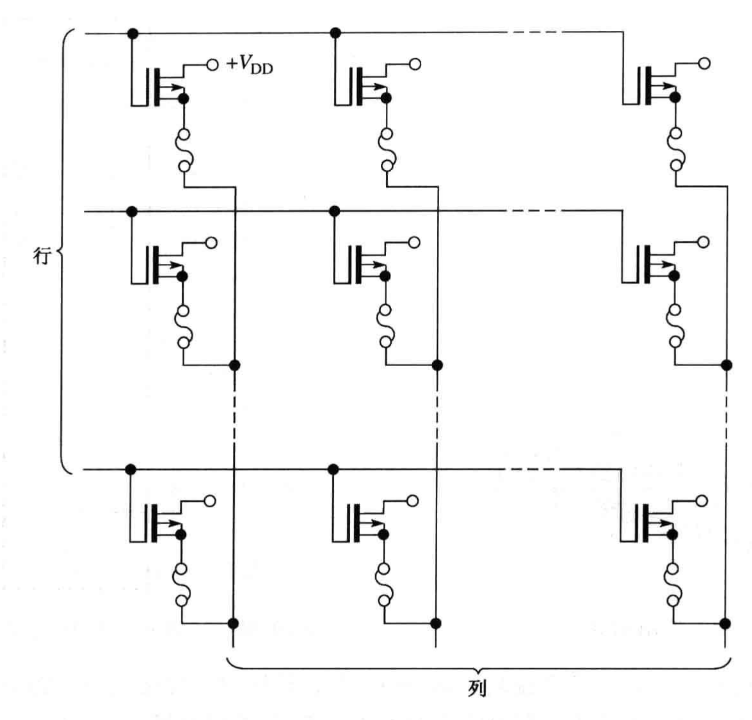


图 10.29　具有熔丝连接的 MOS PROM(所有的漏极共同连接到 V_{DD})

10.4.2　EPROM

EPROM 是可擦除 PROM。和普通的 PROM 不同，存储阵列中现有的程序首先被擦除，然后 EPROM 可以被重编程。

EPROM 使用具有绝缘栅极结构的 NMOSFET。绝缘晶体管的栅极没有电的连接，从而可以在无限长的时间里存储一个电子电荷。这种类型的阵列中的数据位由栅极是否存储电荷来表示。数据位的擦除是移走栅极电荷的过程。

可擦除 PROM 的两种基本类型是紫外线可擦除 PROM(UV EPROM)和电可擦除 PROM(EEPROM)。

UV EPROM　可以通过封装上的透明石英盖来识别 UV EPROM 芯片，如图 10.30 所示。UV EPROM 的场效应管中的绝缘栅极"浮动"在氧化绝缘材料内。编程过程使得电子从浮动栅极中移走。擦除是通过封装顶上的石英窗口，把存储阵列芯片暴露在高密度紫外线辐射下而完成的。存储在栅极上的正电荷在几分钟到一小时的暴露时间后得到中和。

典型的 UV EPROM 由图 10.31 中的逻辑框图表示。其操作是其他不同大小的典型 UV EPROM 的代表。正如逻辑符号所给出的那样，这个芯片具有 2048 个地址($2^{11}=2048$)，每一个具有 8 个位。注意 8 个输出是三态(∇)的。

为了从存储器中读出数据，输出使能输入($\overline{OE}$)必须为低电平，而断电/编程($\overline{CE}/PGM$)输入为低电平。为了擦除存储的数据，芯片通过透明盖暴露于高密度紫外线下。典型的紫外线灯将会在 20～25 分钟内把数据擦除掉。和大多数的 UV EPROM 一样，擦除后所有的位都是 1。正常的环境光线包含真实的紫外波长的光，可以在一段时间内完成擦除。因此，封装上的透明盖必须保持掩盖。

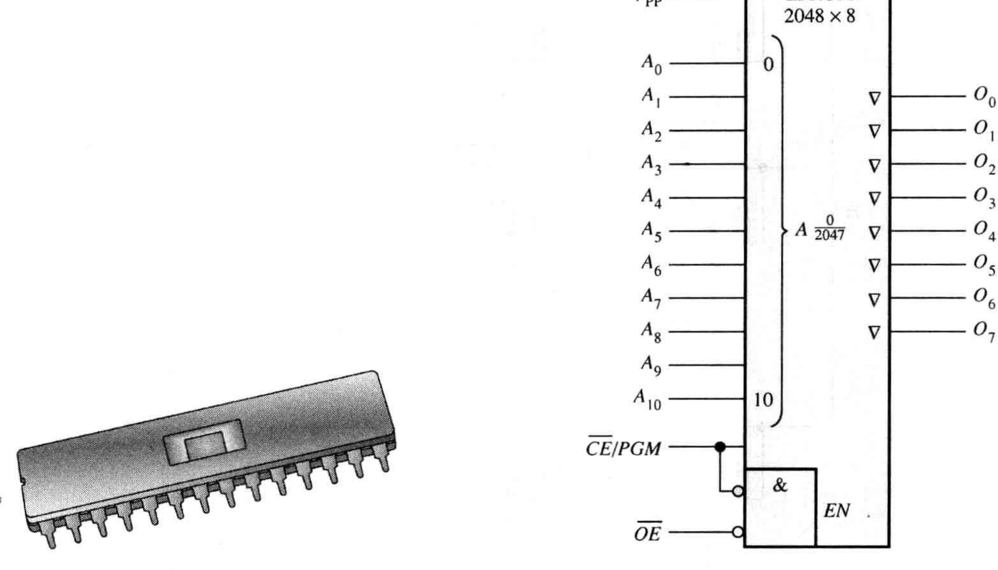


图 10.30　UV EPROM 的封装

图 10.31　2048 ×8 UV EPROM 的逻辑符号

为了对芯片进行编程，一个直流高压加到 V_{PP}上，并且 $\overline{OE}$ 为高电平。将编入给定地址的 8 个数据位加在输出($Q_0 \sim Q_7$)上，而地址在输入 $A_0 \sim A_{10}$上进行选择。接下来，一个高电平脉冲加到 $\overline{CE}/PGM$ 输入。地址可以在任意的顺序下编程。

这个编程过程的时序图如图 10.32 所示。这些信号一般由 EPROM 编程器产生。

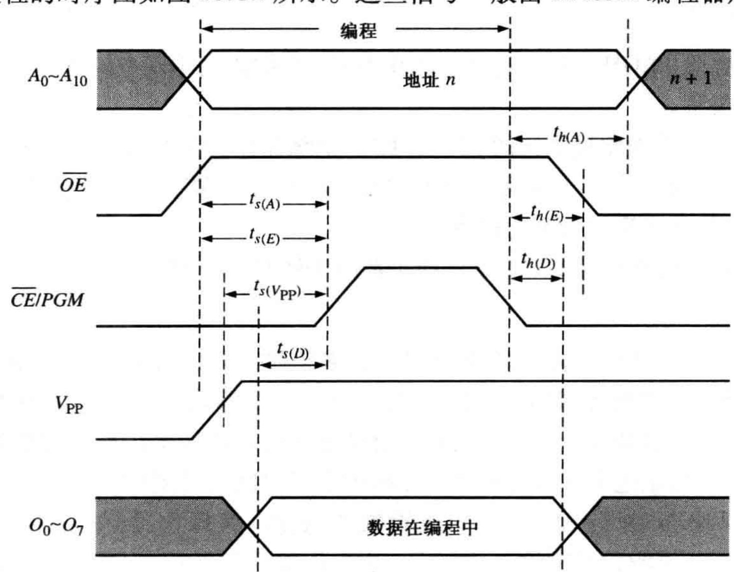


图 10.32　2048 ×8 UV EPROM 编程周期的时序图，指出了关键的建立时间(t_s)和保持时间(t_h)

EEPROM　EEPROM 可以利用电脉冲来擦除和编程。由于它可以电写入和电擦除，所以 EEPROM 可以在重编程的电路中被快速地编程和擦除。

EEPROM 的两种类型是浮栅 MOS 和金属氮化物 - 氧化物硅(MNOS)。浮栅结构中控制栅极上施加的电压允许从浮栅中存储或者移走电荷。

10.4 节　温故而知新

1. PROM 和 ROM 的区别是什么?
2. 擦除以后，一个典型的 EPROM 中的所有位是 1 还是 0?
3. 一个 PROM 的正常运行模式是什么?

10.5　闪存

闪存(flash memory)是高密度的非易失性读/写存储器(高密度意味着较大的位存储容量)，也就是说数据可以在没有电源的情况下永远地存储下来。它们有时候用来取代便携式计算机的软盘或者小容量硬盘。

高密度的意思是芯片上给定的表面区域内可以容纳大量的单元；也就是说，密度越高，给定尺寸的芯片所能存储的位也就越多。在闪存中使用由单浮栅 MOS 晶体管组成的存储单元，使高密度得以实现。数据位的存储依据存储 0 还是 1，对应于浮栅上有电荷或者没有电荷。

10.5.1　闪存单元

闪存中的单个晶体管单元如图 10.33 所示。层叠的栅极 MOS 晶体管由一个控制栅极和一个浮栅及漏极和源极组成。一个足够的电压加在控制栅极上，导致浮栅存储电子(电荷)。当浮栅上有更多电荷时，就存储 0；而当电荷很少或者没有电荷时，就存储 1。若读操作期间加上控制电压，这时出现在浮栅上的电荷的多少取决于晶体管是否导通，以及从漏极到源极的导通电流。

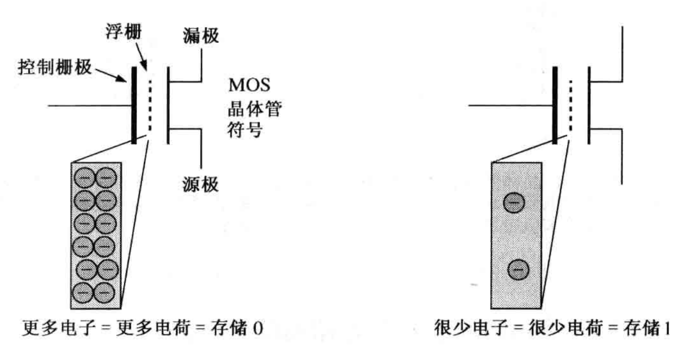


图 10.33　闪存中的存储单元

10.5.2　基本闪存操作

在闪存中有三种主要的操作：编程操作、读操作和擦除操作。

编程　初始时，所有的单元都处在 1 状态，因为在先前的擦除操作中把每个单元的电荷移走了。编程操作在要存储 0 的那些单元的浮栅中加入电子(电荷)，那些存储 1 的单元不加入电荷。在编程期间，施加一个相对于源极的足够的正电压到控制栅极，以此把电子吸引到浮栅上，如图 10.34 所示。一旦编程之后，在没有外部电源的情况下，单元上的电荷可以保留 100 年。

读　在读操作期间，正电压加在控制栅极上。单元浮栅所呈现的电荷量取决于加在控制栅极的电压是否使晶体管导通。如果存储 1，控制栅极的电压就足以使晶体管导通；如果存储 0，晶体管将不会导通，因为控制栅极的电压不足以克服存储在浮栅中的负电荷。把浮栅上的电荷

考虑为一个电压源，对抗读操作期间加在控制栅极上的电压。所以和存储的 0 相对应的浮栅电荷，会阻止控制栅极的电压达到能使晶体管导通的电压阈值；而和存储 1 相对应的较少或者零电荷，则允许控制栅极的电压超过使晶体管导通的电压阈值。

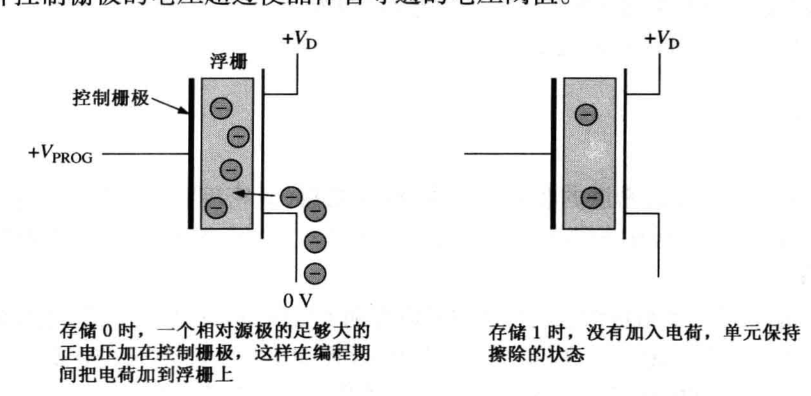


图 10.34 编程操作期间闪存单元存储 0 或者 1 的简单解释

当晶体管导通时，就会有电流从漏极流向单元晶体管的源极。这个电流的出现就表示一个 1，没有电流就表示一个 0。这个基本思想如图 10.35 所示。

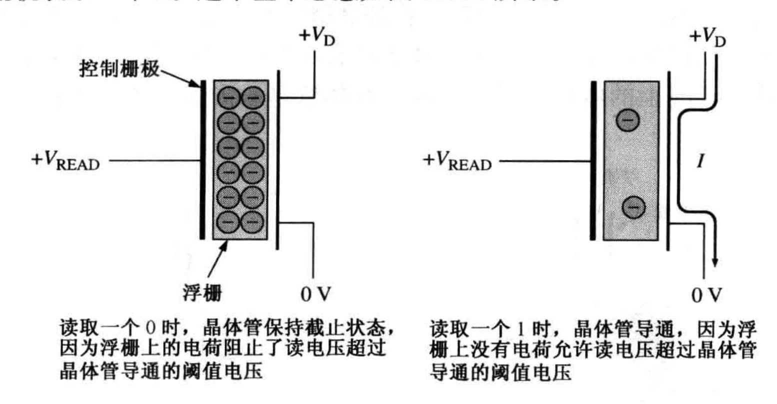


图 10.35 一个阵列中闪存单元的读操作

擦除 在擦除操作期间，电荷被从所有的存储器单元中移走。一个相对于控制栅极的足够大的正电压加在晶体管源极。这和在编程中使用的极性是相反的。此电压从浮栅中吸引电子，并耗尽它的电荷，如图 10.36 所示。闪存总是在重编程之前被擦除。

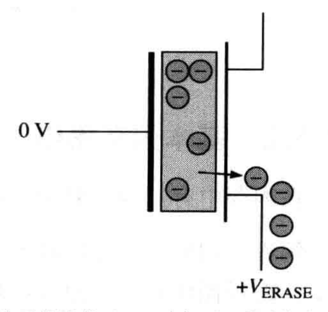


图 10.36 擦除期间从单元中移走电荷的简单解释

10.5.3 基本闪存阵列

闪存单元的简化阵列如图 10.37 所示。每次只访问一条行线。在读操作期间，当一条给定位线中的单元打开(存储 1)时，就会有电流经过这条位线，产生穿越有效负载的电压降。这个电压降使用一个比较器电路和参考电压进行比较，并且产生指示 1 的输出电平。如果 0 被存储，那么在位线中就不会有电流或者有很小的电流，并在比较器输出上产生一个相反的电平。

存储棒是一种存储介质，在物理配置中它使用闪存技术，其尺寸比一条口香糖还要小。典型的存储棒的容量有 128 MB ~ 2 GB，并且作为 PC 卡适配器的配套设备。由于它紧凑的设计，将其用在小型数字电子产品(例如笔记本电脑和数码相机)中是很理想的。

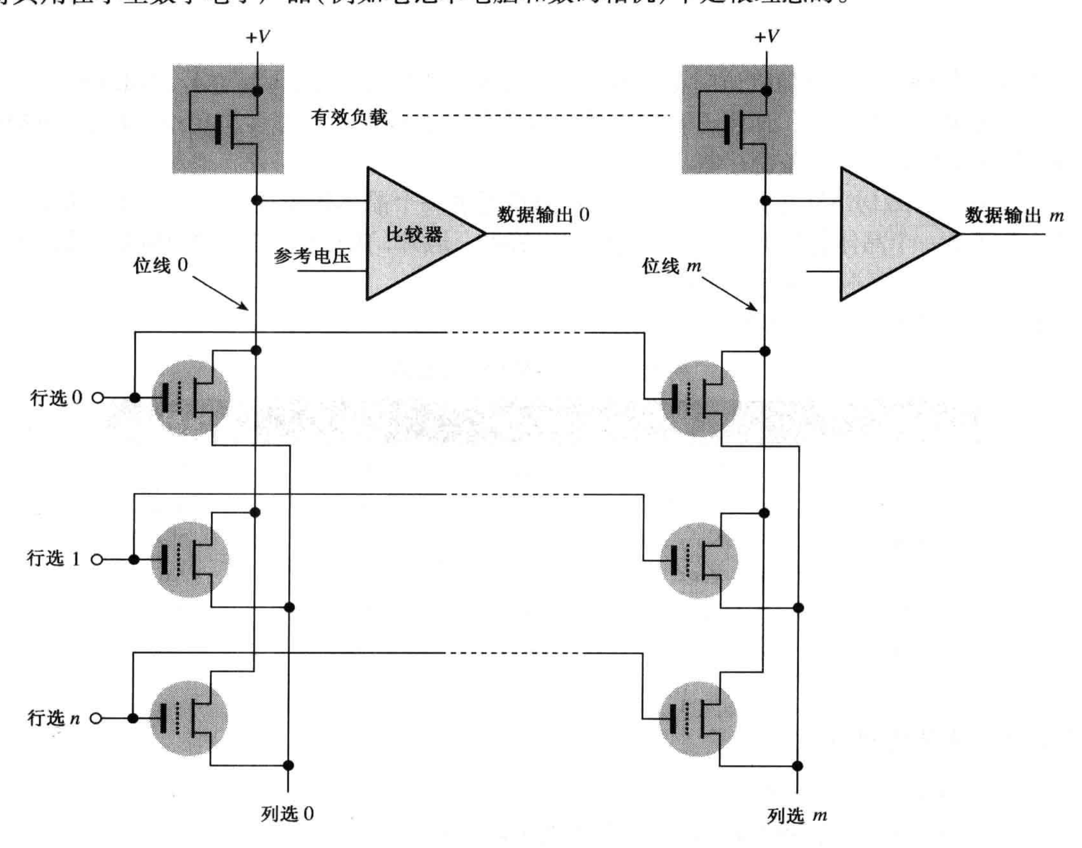


图 10.37　基本闪存阵列

10.5.4　闪存和其他存储器的比较

下面把闪存和所熟悉的其他类型的存储器进行比较。

闪存与 ROM、EPROM 和 EEPROM　只读存储器是高密度、非易失的芯片。但是，一旦编程之后，ROM 的内容就不能改变。同样，初始编程也是费时、费钱的过程。

虽然 EPROM 是高密度、非易失的存储器，但是它只能通过将存储内容移出系统和使用紫外线灯来擦除。它只能使用专门的设备来重编程。

EEPROM 比 ROM 或者 EPROM 具有更加复杂的单元结构，并且它的密度并不是太高，但是可以在存储内容不移出系统的情况下进行重编程。由于它的密度低，每位的成本要比 ROM 或者 EPROM 高许多。

闪存可以很容易地在系统中进行重编程，因为它本质上是一种读/写芯片。闪存的密度可以与 ROM 和 EPROM 相媲美，因为它们都具有单个晶体管单元。闪存(和 ROM、EPROM 或者 EEPROM 一样)是非易失性的，允许在电压断开的情况下永久地存储数据。

闪存与 SRAM　正如所学到的那样，静态随机访问存储器(SRAM)是易失性的读/写芯片。SRAM 需要恒定的电源来保持它所存储的数据。在许多应用中，如果主电源被关闭，则使用备

用电池来防止数据丢失。但是，由于电池故障总是会出现，存储数据的永远保存是不能保证的。因为 SRAM 中的存储单元基本上是由几个晶体管组成的触发器，所以密度相对较低。

闪存也是读/写芯片，但是和 SRAM 不同，它是非易失性的。同样，闪存比 SRAM 具有更高的密度。

闪存与 DRAM 动态随机访问存储器(DRAM)是易失性高密度读/写芯片。DRAM 不仅需要恒定的电源来保存数据，而且还需要经常刷新所保存的数据。在许多应用中，对于 DRAM，必须使用诸如硬盘之类的备份存储。

闪存呈现出比 DRAM 更高的密度，因为闪存单元由一个晶体管组成，并且不需要刷新，而 DRAM 单元是一个晶体管加上一个必须刷新的电容器。通常，闪存要比等价的 DRAM 耗费较少的电量，并且在许多应用中可以用来取代硬盘。

表 10.2 提供了存储器技术比较的总结。

表 10.2 存储器类型的比较

存储器类型	非易失性的	高密度	一个晶体管单元	可以写入系统
闪存	Yes	Yes	Yes	Yes
SRAM	No	No	No	Yes
DRAM	No	Yes	Yes	Yes
ROM	Yes	Yes	Yes	No
EPROM	Yes	Yes	Yes	No
EEPROM	Yes	No	No	Yes

10.5 节 温故而知新

1. 非易失性存储器的类型是什么?
2. 闪存和 SRAM 或 DRAM 相比较，它的优点是什么?
3. 列出闪存工作的 3 个模式。

10.6 存储器扩展

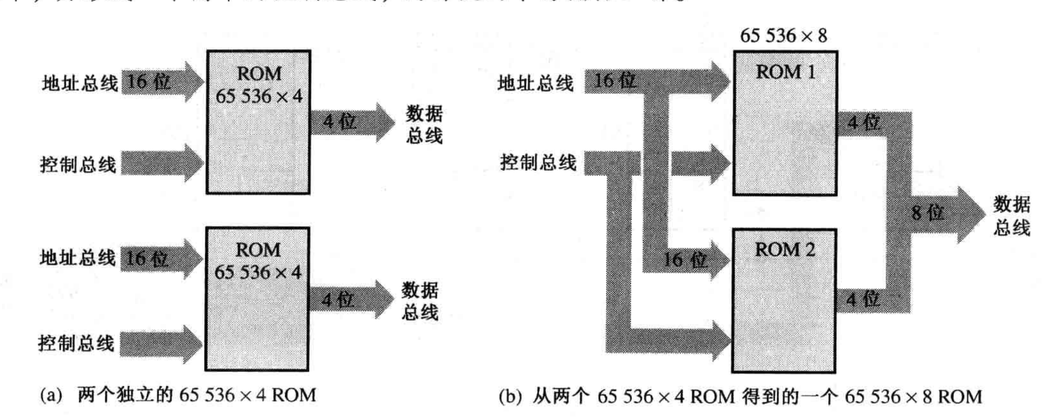
10.6.1 字长扩展

为了增加存储器的字长，必须增加数据总线中位的个数。例如，8 位字长可以通过使用两个存储器来实现，每个存储器都具有 4 位的字，如图 10.38(a)所示。正如在图 10.38(b)看到的那样，16 位地址总线通常连接到两个存储器上，使得组合而成的存储器仍然具有和每一个存储器相同的地址数($2^{16}=65\ 536$)。来自两个存储器的 4 位数据总线组合在一起而形成一个 8 位数据总线。这样，当选中一个地址时，就会在数据总线上产生 8 个位——从每个存储器得到 4 位。例 10.2 给出了详细的 65 536 × 4 到 65 536 × 8 的扩展。

例 10.2 扩展图 10.39 中的 65 536 × 4 ROM(64k × 4)形成一个 64k × 8 ROM。注意“64k”是 65 536 的缩写。为什么不是“65k”? 可能因为 64 也是 2 的幂。

解： 两个 64k × 4 ROM 按照如图 10.40 所示连接在一起。注意 ROM 1 和 ROM 2 中的一个特定地址同时得到访问。4 个位来自 ROM 1 中一个选中的地址，4 个位来自 ROM 2 中的相应地

址，这两个地址并行输出，在数据总线上形成一个 8 位字。同时还要注意芯片使能线 $\overline{E}$ 上的低电平，其形成一个简单的控制总线，同时使两个存储器工作。



(a) 两个独立的 65 536 × 4 ROM　(b) 从两个 65 536 × 4 ROM 得到的一个 65 536 × 8 ROM

图 10.38　两个 65 536 × 4 ROM 扩展为一个 65 536 × 8 ROM，用来解释字长的扩展

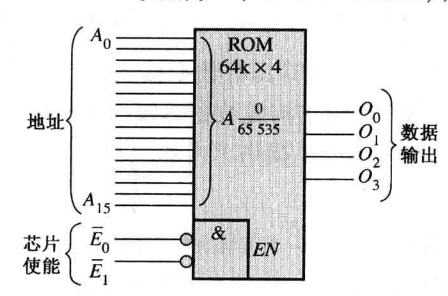


图 10.39　一个 64k × 4 ROM

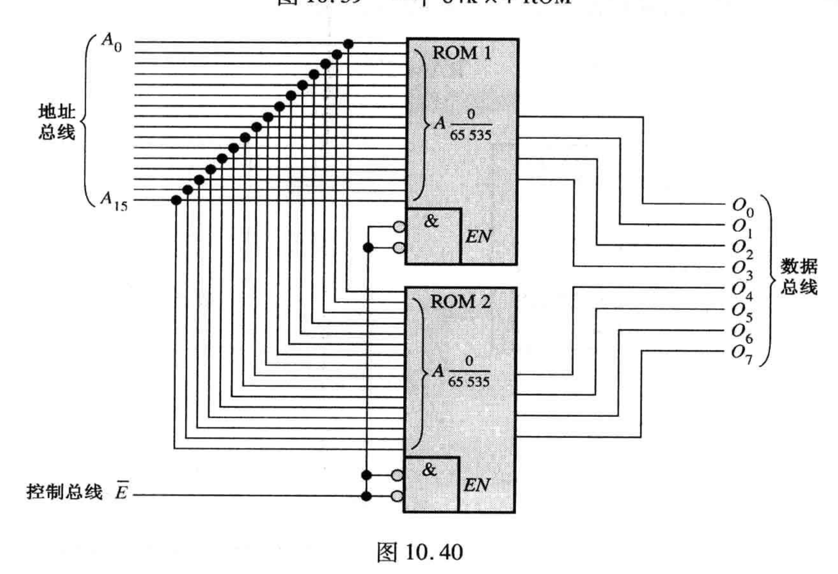


图 10.40

相关问题： 描述怎样将 64k × 1 ROM 扩展到 64k × 8 ROM。

例 10.3　使用例 10.2 中的存储器形成 64k × 16 ROM。

解： 在这种情况下，需要一个存储 65 536 个 16 位字的存储器。需要 4 个 64k × 4 ROM 来完成这个任务，如图 10.41 所示。

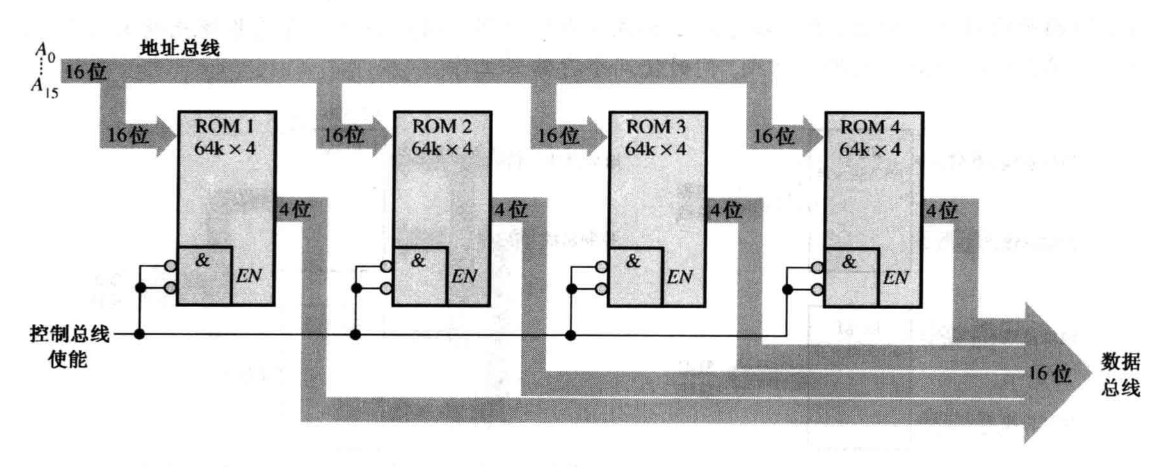


图 10.41

相关问题: 需要多少个64k×1 ROM才能实现如图10.41所示的存储器。

ROM只具有数据输出，但是RAM同时具有数据输入和数据输出。对于RAM(SRAM或者DRAM)的字长扩展，数据输入和数据输出构成数据总线。因为数据输入和数据输出都使用相同的线，所以需要三态缓冲器。大多数RAM提供了内部三态电路。图10.42解释了RAM的扩展，用以增加数据字长。

例 10.4 使用1M×4 SRAM来创建一个1M×8 SRAM。

解: 按照如图10.43所示的简化框图，把两个1M×4 SRAM连接在一起。

相关问题: 使用1M×8 SRAM来创建一个1M×16 SRAM。

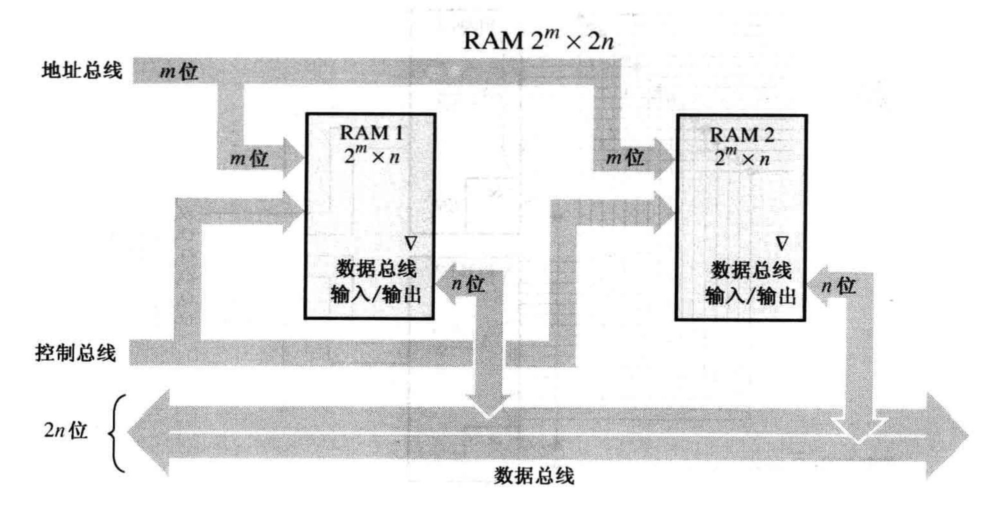


图 10.42 使用两个 $2^m \times n$ RAM形成一个 $2^m \times 2n$ RAM的字长扩展的解释

10.6.2 字容量扩展

当扩展存储器以增加字容量时，地址的数目就增加了。为了实现这个增加，必须增加地址位的个数，如图10.44所示(其中两个1M×8 RAM被扩展成一个2M×8的存储器)。

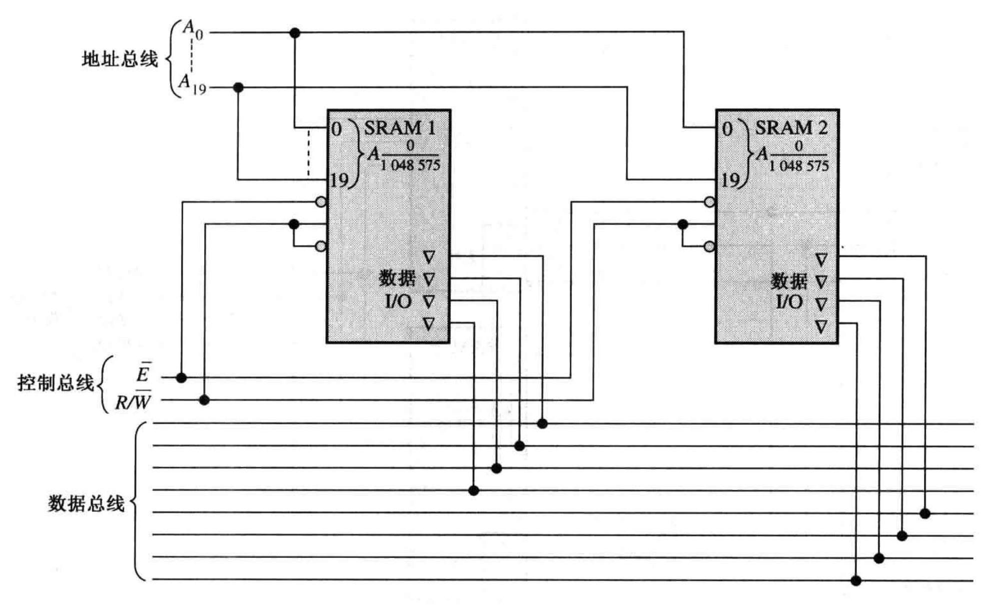
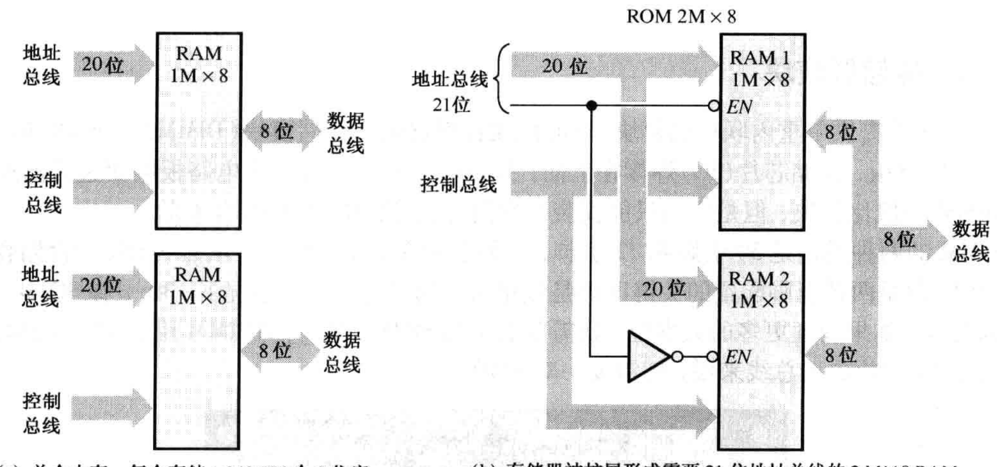


图 10.43



(a) 单个内存，每个存储 1 048 576 个 8 位字　(b) 存储器被扩展形成需要 21 位地址总线的 2 M×8 RAM

图 10.44　字容量扩展的解释

每个独立的存储器都具有 20 个地址位以选择它的 1 048 576 个地址，如图 10.44(b)所示。扩展的存储器具有 2 097 152 个地址，所以需要 21 个地址位，如图 10.44(b)所示。第 21 个地址位用来使能合适的存储芯片。扩展存储器的数据总线保持 8 位的宽度。扩充的细节如例 10.5 所示。

例 10.5　使用 512k ×4 RAM 来实现一个 1M ×4 存储器。

解：扩展寻址通过连接芯片使能($\bar{E}_0$)输入到第 20 个地址位(A_{19})来实现，如图 10.45 所示。输入 $\bar{E}_1$ 用做两个存储器共同的使能输入。当第 20 个地址位(A_{19})为低电平时，RAM 1 就被选中(RAM 2 禁止)，而 19 个较低级的地址位(A_0 ~ A_{18})访问 RAM 1 中的每一个地址。当第 20 个地址位(A_{19})为高电平时，RAM 2 被反相器输出的低电平使能(RAM 1 禁止)，而 19 个低级地址位(A_0 ~ A_{18})访问 RAM 2 中的每一个地址。

相关问题：图 10.45 中 RAM 1 和 RAM 2 的地址范围是多少？

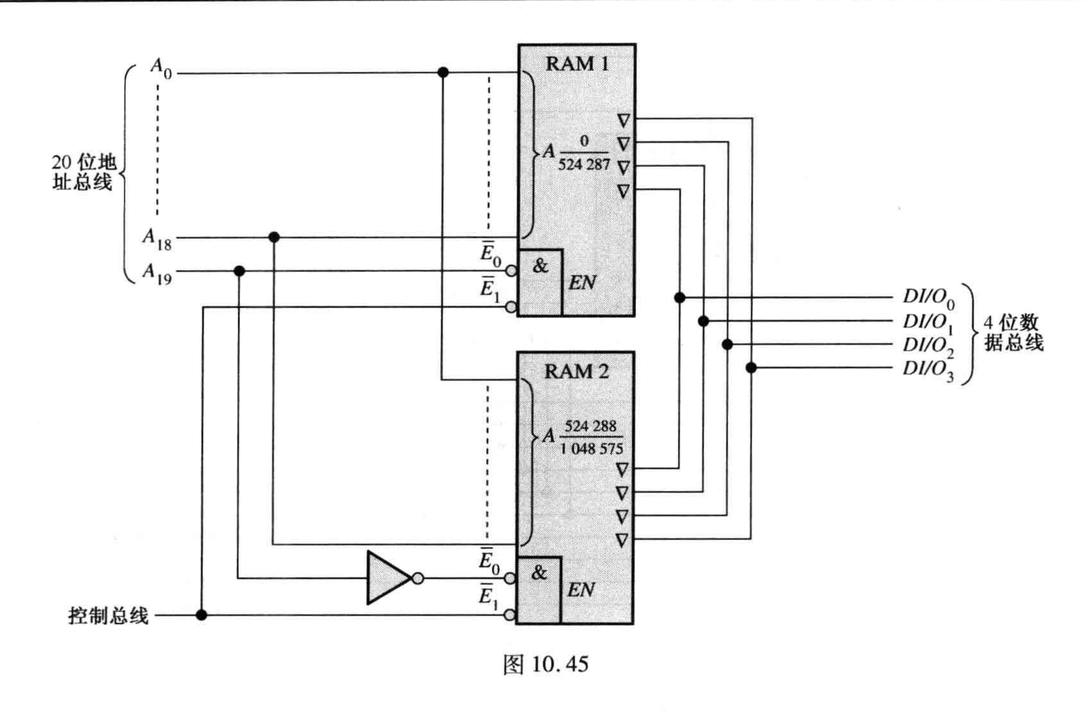


图 10.45

10.6.3 存储器模块

RAM 通常提供单重内联存储模块(SIMM)或者双重内联存储模块(DIMM)。SIMM 和 DIMM 都是小型电路板，存储芯片(IC)焊接在上面，并且输入和输出连接到电路板底面边沿的连接口上。DIMM 一般比较快，但是它们只能安装在专门为它设计的机器中。

SIMM 的两种类别是 30 引脚和 72 引脚，它们展示在图 10.46 中。虽然 SIMM 的存储容量可到 64 MB，但是两种引脚配置的关键区别是数据通路的尺寸。一般情况下，30 引脚 SIMM 用于 8 位数据总线，如果处理更多的数据位，就需要更多的 SIMM。72 引脚 SIMM 用于 32 位数据总线，所以对于 64 位的数据总线来说，就需要一对 SIMM。

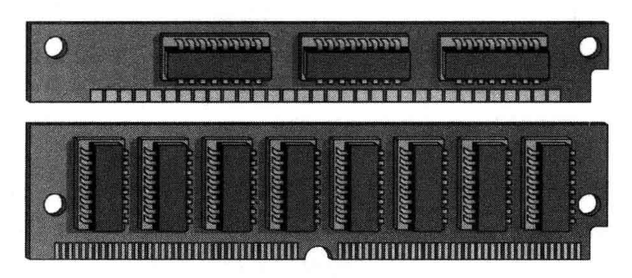


图 10.46 30 引脚和 72 引脚的 SIMM

DIMM 看起来和 SIMM 很相似，但是在物理尺寸上仅有相当微小的增加，结果就增加了存储器的密度。关键区别是 DIMM 把输入和输出分布在 PC 主板的两侧，而 SIMM 只使用一侧。常见的 DIMM 配置为 72 引脚、100 引脚、144 引脚和 168 引脚，适用于 32 位和 64 位的数据通路。一般情况下，DIMM 的容量范围从 4 MB 到 512 MB。

SIMM 和 DIMM 插入系统主板上的插座板上，如图 10.47 所示，一般会有几个插座用于存储器的扩展。当然，SIMM 和 DIMM 的插座是不同的，不可互相交换。

另一种标准的存储模块与DIMM相似，但是具有更高的总线速度，就是RIMM（内存总线内联存储模块）。同样，许多笔记本电脑使用一种DIMM的变异版本，称为SODIMM，它的尺寸更小，有144个引脚和多达256 MB的容量。

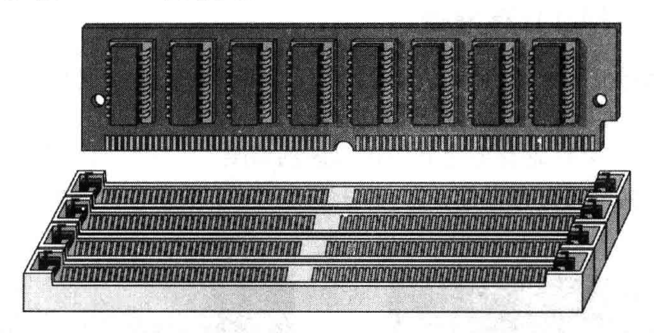


图10.47　插入系统主板插座的SIMM/DIMM



相应提示

存储器组件对静电极度敏感。当处理诸如SIMM和DIMM这样的存储芯片时，可以使用下面的预防措施：

- 在使用前，通过触摸接地的表面或者带上含有高电阻的接地手腕皮带，以便放走身体上的静电。一个方便且可靠的接地方法是交流输出端接地。
- 在准备安装之前，不要把组件从它们的防静电袋子中取出。
- 不要把组件部分放在防静电袋子上，因为只有袋子内部是防静电的。
- 当处理SIMM或者DIMM时，抓住边沿或者金属安装托架。不要触摸主板上的元件或者边沿连接口的引脚。
- 绝对不要在任何类型的表面上滑动元件的任何部分。
- 在工作区域内避免出现塑料、乙烯基、泡沫聚苯乙烯和尼龙材料。

安装SIMM和DIMM时，要遵循以下的步骤：

1. SIMM或DIMM板上的槽口和存储器插座上的槽口对齐。
2. 用力推压模块直到可靠地插在插座上面。
3. 通常，当模块完全插入时，插座两边的锁口会突然锁定位置。这些锁口也可以释放模块，这样模块就可以从插座上取出。

10.6节　温故而知新

1. 要得到一个字的容量为16 k和字长为8位的存储器，需要多少个16 k×1的RAM芯片？
2. 要得到一个字的容量为16 k×8的存储器，需要多少个16 k×1的RAM芯片？
3. SIMM表示什么？
4. DIMM表示什么？
5. RIMM表示什么？

10.7 特殊类型的存储器

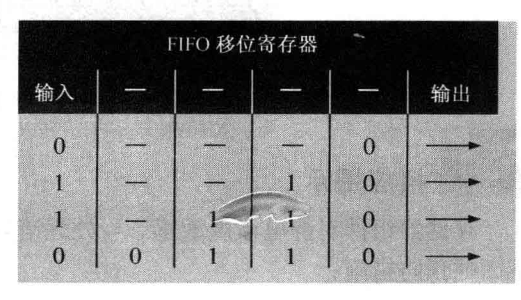
10.7.1 先进先出(FIFO)存储器

这种类型的存储器由移位寄存器的排列组成。术语 FIFO 是指这种存储器的基本操作，即在操作中，首先写入存储器的数据位首先被读出。

常规移位寄存器和 FIFO 寄存器之间的重要区别如图 10.48 所示。在常规寄存器中，数据位仅仅作为新进入的数据位通过寄存器；在 FIFO 寄存器中，数据位即刻穿过寄存器而到达最右边空闲位的位置。

常规移位寄存器

输入	X	X	X	X	输出
0	0	X	X	X	→
1	1	0	X	X	→
1	1	1	0	X	→
0	0	1	1	1	→



FIFO 移位寄存器

输入	—	—	—	—	输出
0	—	—	—	0	→
1	—	—	1	0	→
1	—	1	1	0	→
0	0	1	1	0	→

X=未知数据位

在常规移位寄存器中，数据停留在左边，直到由添加的数据“强迫”它通过寄存器。

— = 空闲位置

在 FIFO 移位寄存器中，数据“陷”入（靠右进入）寄存器。

图 10.48　常规移位寄存器与 FIFO 移位寄存器的比较

图 10.49 是 FIFO 串行存储器的框图。这种特殊的存储器有 4 个串行 64 位数据寄存器和一个 64 位控制寄存器(标记寄存器)。当数据在移入脉冲的作用下进入寄存器时，它们在标记寄存器的控制下自动移向最靠近输出的空位置。数据不能进入被占用的位置。但是，当数据位在移出脉冲的作用下移出时，寄存器中剩余的数据位自动向输出方向移到下一个位置。在异步 FIFO 中，由于使用两个独立的时钟，数据的移出独立于数据进入。

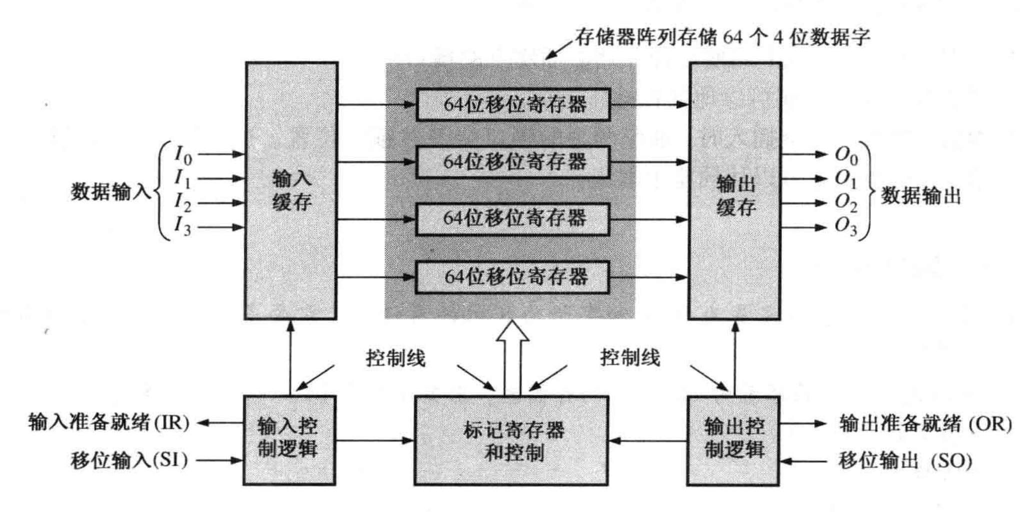
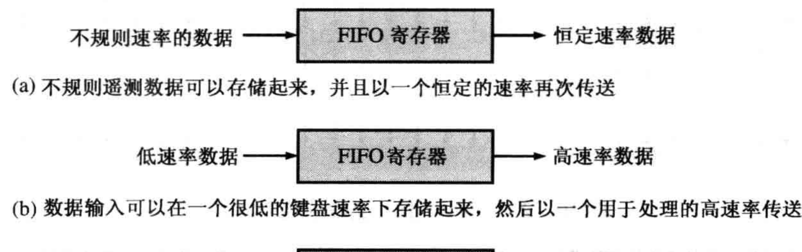


图 10.49　典型的 FIFO 串行存储器的框图

10.7.2　FIFO 应用

FIFO 寄存器的一个重要的应用领域，即两个不同数据速率的系统必须进行通信的情况。数据可以在一种速率下进入 FIFO 寄存器，而在另一种速率下从寄存器中取出。图 10.50 给出了在这些情况下 FIFO 寄存器是如何使用的。



(a) 不规则遥测数据可以存储起来，并且以一个恒定的速率再次传送

(b) 数据输入可以在一个很低的键盘速率下存储起来，然后以一个用于处理的高速率传送



(c) 数据输入可以在恒定速率下存储，然后以脉冲串的形式输出

脉冲串数据 → FIFO 寄存器 → 恒定速率数据

(d) 脉冲串数据可以存储起来，并且重新格式化成恒定速率的数据

图 10.50　典型的 FIFO 串行存储器的框图

10.7.3　后进先出(LIFO)存储器

LIFO(后进先出)存储器经常出现在涉及微处理器和其他计算系统的应用中，它允许数据存储下来，然后以相反的顺序调用，也就是最后存储的数据字节是最先取出的数据字节。

寄存器堆栈　LIFO 存储器一般是指下推堆栈。在一些系统中，它一般由如图 10.51 所示的寄存器组来实现。一个堆栈可以由任意数目的寄存器组成，但是顶部的寄存器一般称为栈顶。

1　栈顶
2
3
第 n 个寄存器

图 10.51　寄存器堆栈

为了解释这个原理，一个字节的数据被并行推入到栈顶上。每一个相继的字节都把前一个字节推入到下一个寄存器中。这个过程如图 10.52 所示。注意，新的字节总是推入到顶部寄存器上，并且前一个存储的字节被推入栈的更深一层。下推堆栈这个名称就来自于这个特点。

数据字节以相反的顺序被取出。最后进入的数据字节总是位于栈顶，所以当这个数据从堆栈中取出时，其他的字节就会跳到高一级的位置。这个过程如图 10.53 所示。

RAM 堆栈　另一种用在基于微处理器的系统中实现 LIFO 存储器的方法就是把 RAM 的一部分配置成堆栈，而不是使用一组专门的寄存器。正如所看到的那样，对于一个寄存器堆栈，数据从一个位置向上或者向下移动到下一个相邻位置。在 RAM 堆栈中，数据本身并不移动，但是栈顶在一个寄存器的控制下移动，这个寄存器称为堆栈指针。

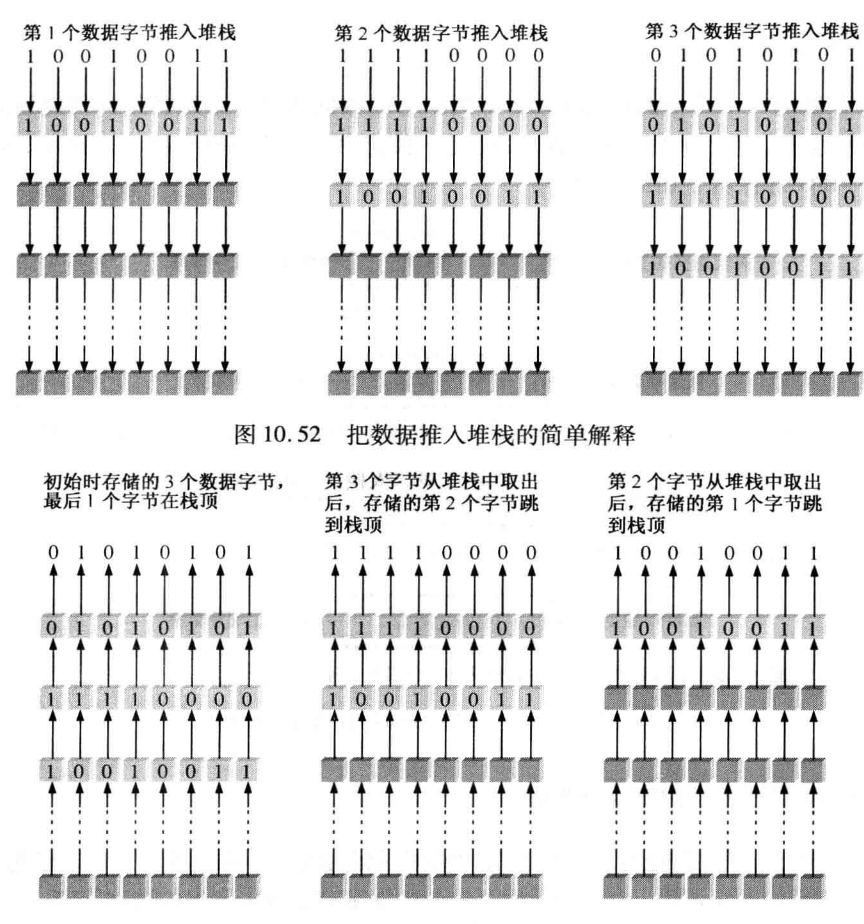


图 10.52　把数据推入堆栈的简单解释

图 10.53　把数据从堆栈取出的简单解释

考虑一个以字节组织的随机存储器，也就是每个地址包含8个位，如图10.54所示。例如，二进制地址0000 0000 0000 1111在十六进制中可以写成000F。一个16位地址的最小十六进制数值为0000_{16}，而最大数值为$FFFF_{16}$。使用这种表示方法，一个64 kB字节的存储阵列可以表示为如图10.54所示的阵列。最低存储地址是0000_{16}，而最高内存地址为$FFFF_{16}$。

现在，考虑留出RAM的一部分用做堆栈。堆栈指针这个特殊的独立寄存器包含栈顶的地址，如图10.55所示。一个4位十六进制表达的数用来表示二进制地址。在图中，为了解释的目的，选择了一个地址。

现在看看数据是怎样被推入堆栈的。堆栈指针初始时位于地址$FFEE_{16}$处，就是如图10.55(a)所示的栈顶。堆栈指针随后被减去(减少)2，到达地址$FFEC_{16}$。这就从栈顶移到了较低的存储地址，如图10.55(b)所示。注意，栈顶和固定寄存器堆栈的栈顶不一样，它不

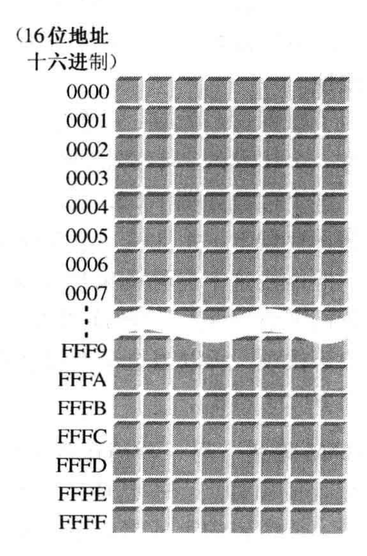


图 10.54　一个64 kB的用十六进制表示16位地址的存储器的示意图

是固定不变的，而是随着数据的存储在 RAM 中向下移动(移向较低的地址)。图 10.55(b)给出了两个字节(一个数据字)随后被推入堆栈中。在存储数据字之后，栈顶位于 $FFEC_{16}$。

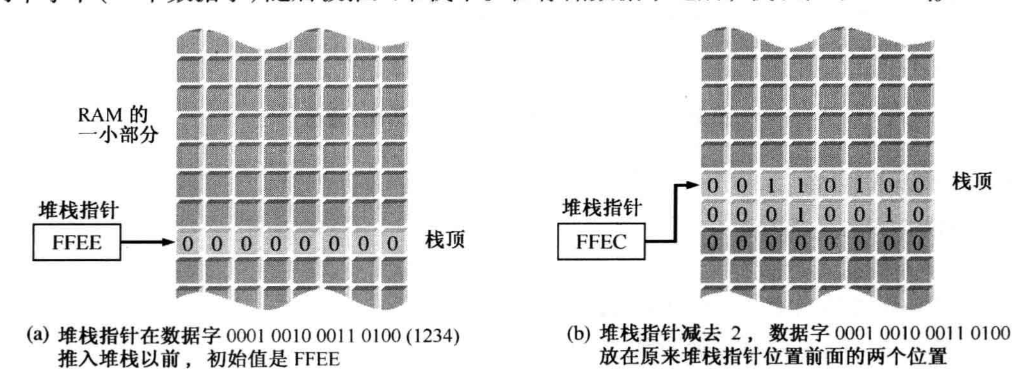


图 10.55　RAM 堆栈的入栈操作的解释

图 10.56 给出了 RAM 堆栈的出栈(POP)操作。堆栈中最后存入的一个数据首先被读出。位于地址 $FFEC_{16}$处的堆栈指针增加 2，到达地址 $FFEE_{16}$，出栈操作的执行如图 10.56(b)所示。记住读操作时，RAM 是非破坏性的，所以数据字在出栈操作之后仍然保持在存储器中。只有当写入新的字并覆盖数据字时，这个数据字才会被破坏。

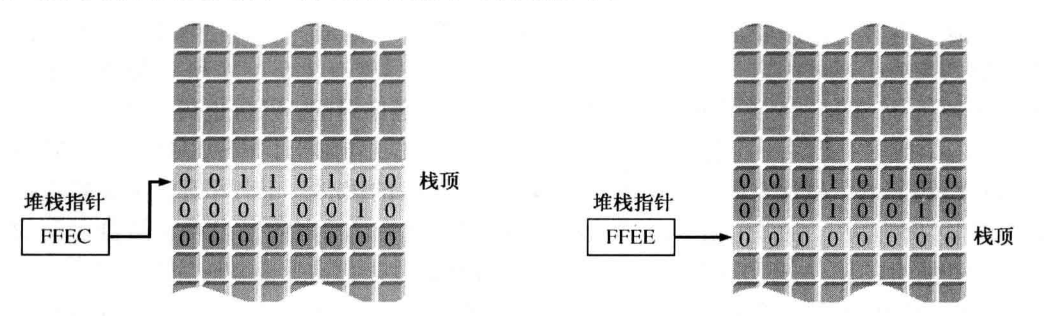


图 10.56　RAM 堆栈的出栈操作的解释

RAM 堆栈可以有任意的深度，这取决于为实现这个目的所分配的连续的存储地址数。

10.7.4　CCD 存储器

CCD(charge-coupled device，电荷耦合器件)存储器以电容上的电荷作为存储的数据。然而，和动态 DRAM 不同，其存储单元不包括晶体管。CCD 的主要优点是高密度。

CCD 存储器由一行行较长的半导体电容(称为通道)组成。存储在电容上的电荷较少时，就是 0，存储的电荷较多时，就是 1，数据串行地进入通道。这些电荷包在时钟信号的作用下，随着更多数据的存入而沿着通道移位。

和 DRAM 一样，电荷必须定期刷新。通过把电荷包串行移位到刷新电路中来完成这个过程。图 10.57 给出了 CCD 通道的基本概念。因为数据在通道中串行移位，所以 CCD 存储器具有相对较长的存取时间。CCD 阵列用在一些现代摄像机中，以光感应电荷的方式来捕捉视频图像。

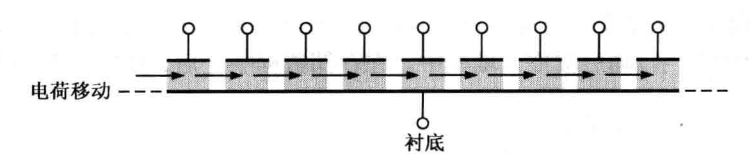


图 10.57 一个 CCD(电荷耦合器件)通道

10.7 节 温故而知新

1. FIFO 存储器是什么?
2. LIFO 存储器是什么?
3. 解释一个存储器中堆栈的入栈操作。
4. 解释一个存储器中堆栈的出栈操作。
5. 术语 CCD 表示什么?

10.8 磁和光存储

10.8.1 磁存储

磁性硬盘 计算机使用硬盘作为外部大容量存储介质。硬盘是一个铝合金或玻璃和表面涂有磁性材料的陶瓷组成的固定“底盘”。硬盘驱动器的尺寸尽管有 2.5 英寸和 1.75 英寸的，但是主要有两种尺寸，即 5.25 英寸和 3.5 英寸。硬盘驱动器是密封的，以防止灰尘进入。

通常，两个或多个磁片一个接一个地堆放在一个公共轴或转轴上，转轴以每秒几千转的速度带动磁片旋转。每个磁片之间分离以允许对读/写磁性头的读/写，读/写磁性头安装在传动臂上，如图 10.58 所示。在每个磁片的两边都有读/写磁性头，因为数据记录在磁盘表面的两边。传动臂和所有的读/写磁性头同步，以保证它们在此磁片表面高速移动时完全对齐，高速移动时读/写磁性头和磁片之间的间距仅有很小的 1 毫米。一个很小的灰尘颗粒将会导致磁头的“破碎”，从而造成磁片表面的损坏。

基本的读/写头工作原理 硬盘是一个随机读写设备，因为它可以按照任何次序在存储数据的任何地方获取数据。如图 10.59 所示，给出了磁盘表面读/写操作的一个简单示意图。磁片表面磁性区域的方向或极性由产生的磁通量线(磁场)控制，线圈内流过的电流的方向确定写磁头产生的磁通量线的方向。磁通量使得沿磁场方向的磁片表面的一个小点区域极化。这个极化的小点区域表示二进制数 1，相反极性的小点区域表示 0。一旦磁片表面的小点区域得到有极性的磁化，则数据保存下来，直到一个极性相反的磁场写入并覆盖它。

当磁表面经过读头时，磁化的小点区域在读头处产生磁场，使得线圈中产生脉冲电压。

这些脉冲电压的极性取决于被磁化的小点区域的极性方向，也就给出了存储的位是 1 还是 0。读头和写头通常组合在一起，形成一个器件。

硬盘的格式化 硬盘由磁道和扇区组成或格式化成磁道和扇区，如图 10.60(a)所示。每个磁道分为若干个扇区，每个磁道和扇区有操作系统使用的物理地址，指出特定的数据记录。典型的硬盘有几百到上千个磁道，具有的容量多达 750 GB。如在图中看到的，磁道/扇区的数目是一个固定的值，外部扇区比内部扇区使用更多的表面区域。磁片上磁道和扇区的排列称为格式化。

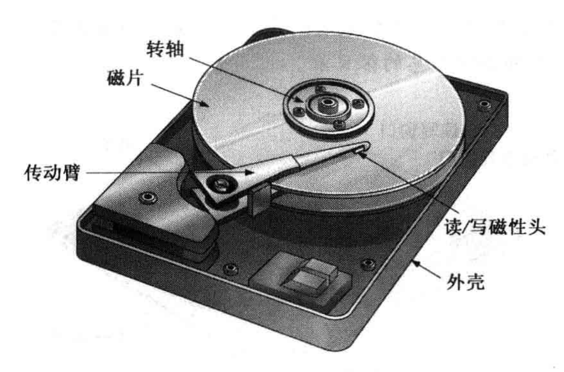




图 10.58　硬盘驱动器

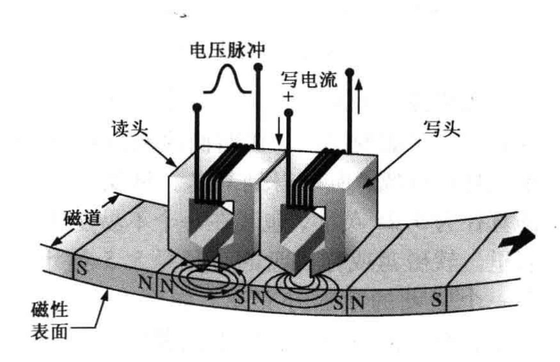


图 10.59　简单的读/写头的操作

图 10.60(b)解释了一个硬盘的堆栈。硬盘驱动器中堆栈磁片的数目不全相同，但是最少有两个相同。在每个磁片中，相对应的磁道都相同，它们的集合称为一个柱面，如图中所示。

硬盘的工作　几个基本参数确定了一个给定的硬盘驱动器的工作。搜索运行就是读/写头移向所需磁道的过程。搜索时间就是这个工作过程的平均时间。典型的硬盘的平均搜索时间为几个毫秒，并且依据于特定的硬盘驱动器。

传输延迟周期是指，一旦读/写头被定位在所需磁道后，所需扇区转到读/写头下面的时间。最坏的情况是，所需扇区刚刚通过读/写头，并且离开它。这时这个扇区必须旋转几乎一周才能回到读/写头的位置。显然，传输延迟时间取决于磁片旋转的速度常数。磁片旋转的速度随不同的磁片驱动器而不同，但是典型的有 3600 转/每分钟(rpm)、4500 rpm、5400 rpm 和 7200 rpm。某些磁片驱动器旋转速度达到 10 033 rpm，平均的传输延迟周期小于 3 ms。

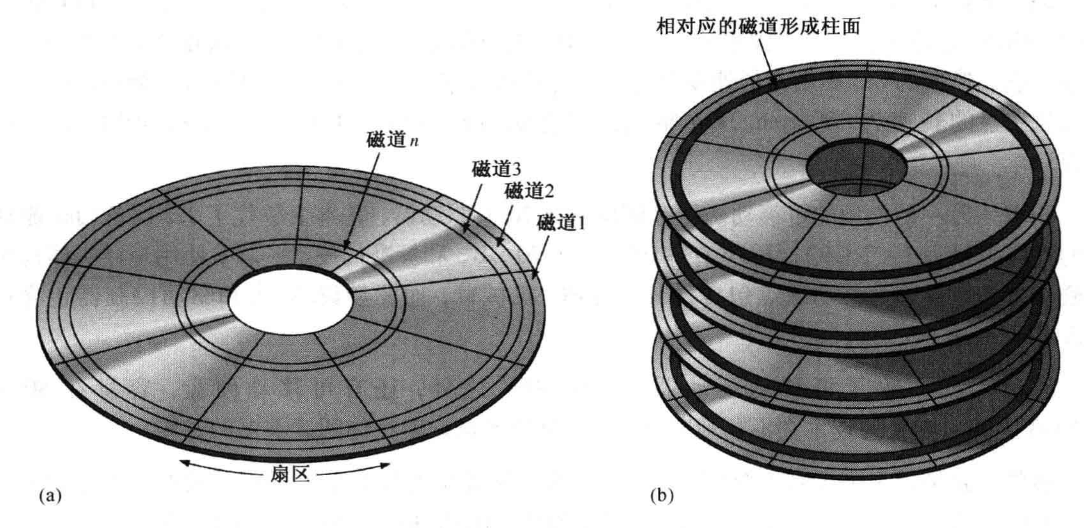


图 10.60　硬盘的排列和格式化

平均寻道时间与平均传输延迟周期的和就是磁盘驱动器的读/写时间。



计算机小知识

数据以文件的形式存储在硬盘驱动器中。保持文件位置的磁道是硬盘管理设备驱动器(有时作为硬盘驱动器 BIOS)的任务。设备驱动器和计算机操作系统可以访问两个表，以保持文件和文件名的磁道。第一

个表称为FAT(文件分配表)。FAT给出特定文件的分配和保持开放扇区和坏扇区的记录。第二个表是具有文件名、文件类型、时间和创建日期、开始簇数和有关文件的其他信息的根目录。

软磁盘 软磁盘是一种很古老的技术，因为它由一个能变形的、在两面涂有磁性物质的聚酯材料组成，所以得到其名。早期的软磁盘的直径为5.25英寸，包装在一个不易变形的外套里。软磁盘或称磁盘的直径是3.5英寸，装在一个坚硬的塑料外套里，如图10.61所示。一个装有弹簧的关闭器覆盖在读写窗口上，并且在插入磁盘驱动器以前保持关闭。一个金属插座上有一个孔，以便让磁盘放在中间，并且在保护外套里面旋转。显然，软磁盘是可移动磁盘，而硬盘不是。软磁盘除了磁道和扇区的数目不同以外，和硬盘相似，格式化以后可以得到相应的磁道和扇区。高密度、容量为1.44 MB的软磁盘每面有80个磁道、18个扇区。

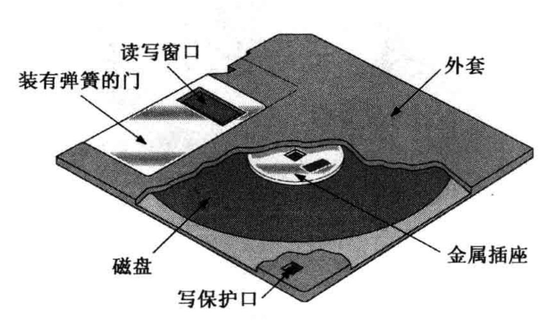


图10.61 3.5英寸的软磁盘



计算机小知识

磁带是磁盘的一个替代物，因为其每个位的成本很低。尽管它的密度比磁盘驱动器要低，但是磁带可以使用的表面积很大。最大容量的磁带介质通常和可以得到的最大磁盘驱动器的容量是同一个级别(大约1 TB, 2007年)。磁带曾经在价格上比磁盘具有更多的优点，使得它成为一个受欢迎的产品，尤其用于备份的目的，而且它是可移动的介质也很重要。

Zip Zip驱动器是可移动磁性存储设备，是替代有限容量的软磁盘的一种类型。和软磁盘一样，Zip磁盘是安装在一个固定外壳里的一个易变形的磁盘，它的尺寸和软磁盘大约相等，但是厚一点。典型的Zip驱动器的速度要比软磁盘的速度快得多，因为它的转速为3000 rpm，与此相比软磁盘的转速为300 rpm。Zip驱动器的存储容量多达250 MB，是1.44 MB的软磁盘的173倍。

REV 另一种类型的可移动磁性存储设备是REV驱动器，基本上替代了Jaz设备(Jaz驱动器的存储容量为1～2 GB)。REV存储35 GB或70 GB，和标准的硬盘一样，使用悬浮头读写旋转磁片上的数据。可移动磁盘包含磁片、转轴、机械臂，而驱动读/写头和驱动控制器包含在REV驱动器中。

移动硬盘 除了可移动驱动器Zip和REV之外，还有可移动硬盘，容量在80～250 GB之间。相关的技术在不断地变化，更先进技术也许就在你阅读本书的时候。

磁带 磁带用来为大容量存储设备备份数据，通常在磁带上是串行读写数据而不是随机访问，因此使用速度比磁盘要慢。有几种类型的磁带，包括QIC、DAT、8 mm和DLT。

QIC是quarter-inch cartridge(四分之一的合式磁带机)的缩写，看上去很像内部有两个卷轴的合式磁带机。QIC的各种标准有36～72磁道，存储容量从80 MB到1.2 GB。最近在Travan标准的革新下，磁带的长度和宽度都有所增加，允许的存储容量达到4 GB。QIC磁带驱动器使用的读/写头具有单个写头，在它的两边有读头。这样磁带可以在两个方向运行，同时磁带驱动器检查刚写入的数据。在录写模式下，磁带移动通过读/写头的速度约为100英寸/秒，如图10.62所示。

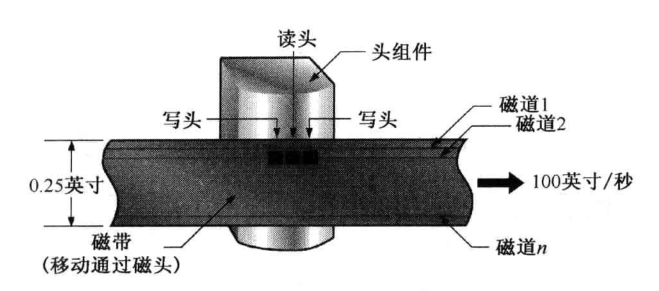


图 10.62 QIC 磁带

DAT 是 digital audio tape(数字音带)的缩写,使用称为螺旋扫描录音的技术。DAT 提供的存储容量的范围达 12 GB,但是它比 QIC 的成本要高。

磁带格式的第 3 种——8 mm 磁带原来用于视频领域,但是已经被计算机领域采纳,作为存储大容量计算机数据的一个可靠方法。8 mm 磁带与 DAT 相似,但是提供的容量多达 25 GB。

DLT 是 digital linear tape(数字线性磁带)的缩写。DLT 是一个半英寸宽的磁带,比 8 mm 磁带宽 60%,当然是标准 QIC 的两倍宽。DLT 的磁带驱动器机械的工作方式和其他系统不同,以便最大限度地减少磨损。DLT 提供所有磁带格式的最高存储容量,多达35 GB。

10.8.2 磁-光存储

顾名思义,磁-光(MO)存储设备使用磁和光的综合技术。一个磁发电-光存盘和磁盘相似,可以格式化成磁道和扇区。

标准的磁盘和 MO 盘的基本区别是:MO 盘上使用的磁涂层需要读/写头去改变它的磁极性。因此,周围温度对 MO 盘的影响非常稳定,数据不会变化。为了写入一个数据位,一个高能量的激光束聚焦在磁盘的一个很小的斑点上,使得这个小斑点的温度上升,达到称为居里(Curie)点的温度(大约 200℃)。一旦被加热,小斑点上可以随意改变方向(极化)的磁性粒子就由于写头产生的磁场而得到改变。使用比写操作功率要小的激光束从磁盘上读取信息,利用克尔(Kerr)效应,即反射回来的激光束随磁性粒子极化的方向不同而变化。一种极化方向的小斑点表示 0,另一种相反方向的表示 1。基本的 MO 操作如图 10.63 所示,给出了磁盘区域的一个小截面。

10.8.3 光存储

CD-ROM 基本只读存储器压缩光盘的直径为 120 mm,即有三个涂层的三明治形式:底部是一个碳酸盐塑料层,一个很薄的铝片用于反射,顶部是一个磁漆层用做保护层。CD-ROM 光盘格式化为单个螺旋磁道,具有 2 kB 序列的扇区,容量为680 MB。数据在出厂前以微小的凹痕形式预先录制在光盘上,这些凹痕称为深坑(pit),围绕这些深坑的区域称为陆地(land)。深坑被压制在塑料层上,不能擦除。

一个 CD 播放机用一个低功率的红外激光束读取旋转磁道上的数据,如图 10.64 所示。数据如图所示以深坑和陆地的形式出现。来自陆地的反射光线和来自深坑的反射光线的相位相差 180°。随着磁盘的旋转,狭窄的激光光束照射在不同长度的深坑和陆地上,光敏二极管检测反射回来的光线的差别。结果产生沿着磁道对应于深坑和陆地的一系列的 1 和 0。

(a) 没有录制过的磁盘

(b) 写：高功率激光束加热小斑点，使得此行粒子随电磁场排列

(c) 读：低功率激光束反射回反相极性的磁粒子，并且光束向极性方向移动。如果极性磁粒子的磁性方向没有反向，那么光束方向不变

(d) 擦除：当电磁场反向、大功率激光束加热小斑点时，使得磁粒子恢复原先的极性

图 10.63　磁-光存储盘的基本原理

WORM　一次写入/多次读取(Write Once/Read Many, WORM)是一种光存储器，它可以一次写入数据，此后数据不能擦除，但是可以多次读取。为了写入数据，使用一个低功率的激光在磁片表面烧制深坑。通过烧制和没有烧制的区域来表示 1 和 0。

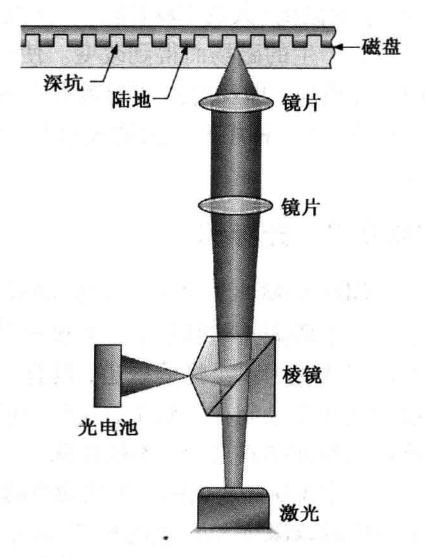


图 10.64　从 CD-ROM 读取数据的基本操作

CD-R　这是一种基本的 WORM 类型。区别是可刻录 CD 允许多次写入磁盘的不同区域。CD-R 磁盘有一个和 CD-ROM 一样的螺旋磁道，除了替代机械地按动光盘上凹痕来表示数据之外，CD-R 光盘使用一束激光把光盘上的微小斑点烧制成一个有机染色表面。读取时，用激光束加热到超过临界温度，烧制过的小斑点比没有烧制过的地方反射较少的光线。因此，通过烧制过和没有烧制过的区域来表示 CD-R 光盘上的 1 和 0，而在 CD-ROM 光盘上是由深坑和陆地来表示 1 和 0。与 CD-ROM 相同，数据一旦写入就不能擦除。

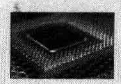
CD-RW　可写入光盘可以用来读取和写入数据。替代 CD-R 中的染色基底录制层，CD-RW 通常使用一种具有特殊性质的晶状合成物。当加温到某个温度时，冷却后成为晶状物；但是当

它加热到某个更高的温度时，就会熔化并且冷却以后变得非晶状。为了写入数据，聚焦的激光束将材料加热到熔化温度，结果形成非晶体状态。产生的非晶状的区域比晶状区域反射较少的光线，这样就允许读操作检测 1 和 0。通过加热非晶状区域到临界温度，低于熔化温度就可以擦除数据，高于熔化温度会造成非晶状材料回到晶状材料。

DVD-ROM　原始的 DVD 是 Digital Video Disk（数字视频盘）的缩写，但是最后表示为 Digital Versatile Disk（数字多用途盘）的缩写。与 CD-ROM 相同，DVD-ROM 数据预先存储在光盘中。然而，深坑的尺寸比 CD-ROM 要小，允许在一个磁道上存储更多的数据。CD-ROM 和 DVD-ROM 之间的主要区别即 CD 是单面的，而 DVD 两面都有数据。除了 DVD 是双面的之外，它在主要数据层上使用有透明数据层的多层盘片，从而提供几十千兆容量的存储。为了访问所有的层，激光束需要重新一层层地聚焦。

10.8 节　温故而知新

1. 列出主要磁存储的类型。
2. REV 的存储容量是多少？
3. 一般情况下，磁盘是如何组成的？
4. 如何从磁-光盘上读取和写入数据？
5. 列出光存储的类型。



数字系统应用

安全系统 第二部分

图 10.65 给出了安全系统的框图。系统由安全密码逻辑、存储器、密码选择逻辑和键盘组成。安全密码逻辑在第 9 章已经设计过了，存储器和密码选择逻辑在应用实践中集中介绍。

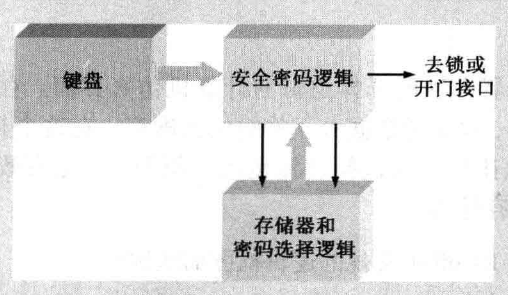


图 10.65　安全系统的框图

存储器和密码选择逻辑

图 10.66 给出了存储器和密码选择逻辑的逻辑框图。在这个系统中的存储器是一些 DIP 开关的集合，DIP 开关的集合用于输入 4 位密码。一开始，按下#键为系统设置第一个密码，该密码即为移位寄存器预置的模式（1000）。初始时，第一个密码的 4 位数由移位寄存器的 Q_0 输出的高电平选中，开启 4 个与门 $A_1 \sim A_4$。随着每个数码在键盘上输入，安全密码逻辑中移位寄存器的 1 在时钟的作用下移位，顺序开启 4 个与门组。结果，安全密码的 BCD 数字顺序出现在输出中。在安全密码逻辑中，每一个密码与键盘输入的时钟相比较，如第 9 章讨论的那样。

1. 初始时，图 10.66 中移位寄存器的输出 Q 的状态是什么？

2. 3 个时钟脉冲之后，移位寄存器的输出 Q 的状态是什么？

3. 解释或门的用途。

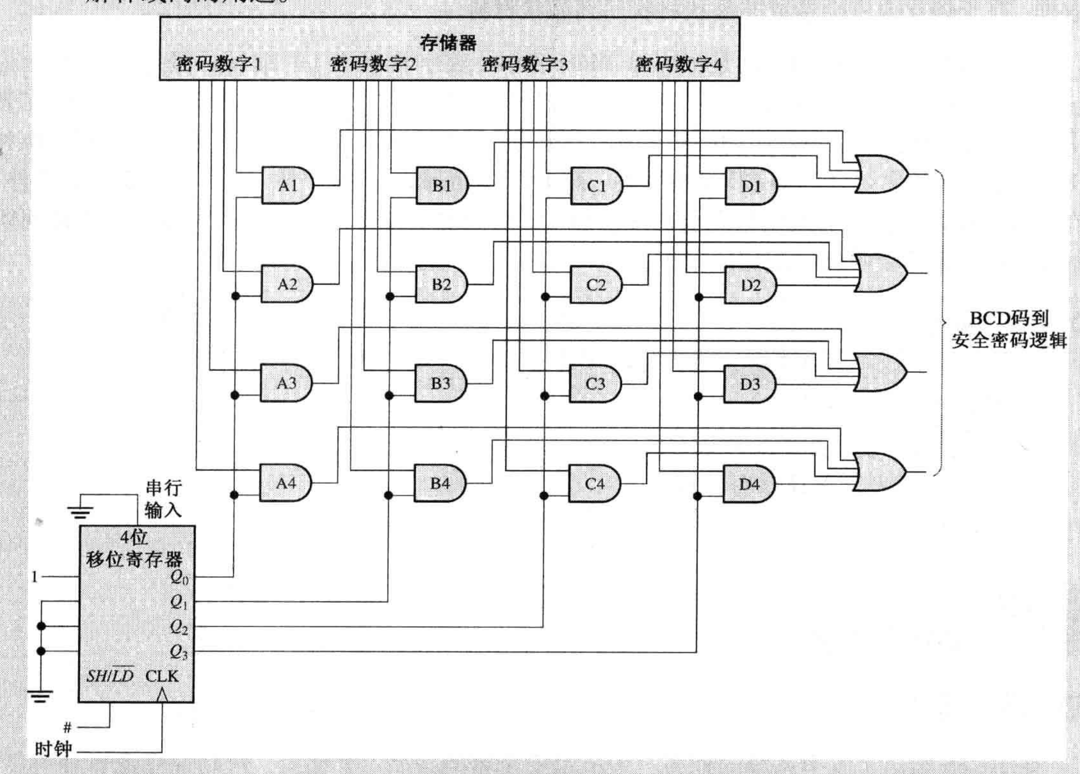


图 10.66 存储器和密码选择逻辑的框图

计算机仿真

图 10.67 给出了存储器和密码选择逻辑的计算机仿真界面。存储器由 4 个 DIP 开关实现。这是一个机械存储器，存储的数据由开关的位置确定。在这个具体的应用中，用户必须为存储器设置密码。为此让开关来选择设置是最方便的，也是有效的输入和存储密码的方法，省去了所需的半导体存储器。

4. 描述图 10.67 和图 10.66 中常规的逻辑框图的区别？

5. 电阻包的用途是什么？

WWW 打开在线资源中系统应用文件夹的文件 SAA10。使用 Multisim 软件运行存储器 1 和密码选择逻辑的计算机仿真程序，并且观察运行结果。

完整的安全密码系统 图 10.68 给出了完整的安全密码系统的框图。存储器和密码选择逻辑连接到安全密码逻辑和给出的键盘。

WWW 为安全密码逻辑、存储器和密码选择逻辑连接 Multisim 电路，运行此仿真电路。

把知识用于实践

解释如何修改存储器和密码选择逻辑使之成为 5 位密码的电路。

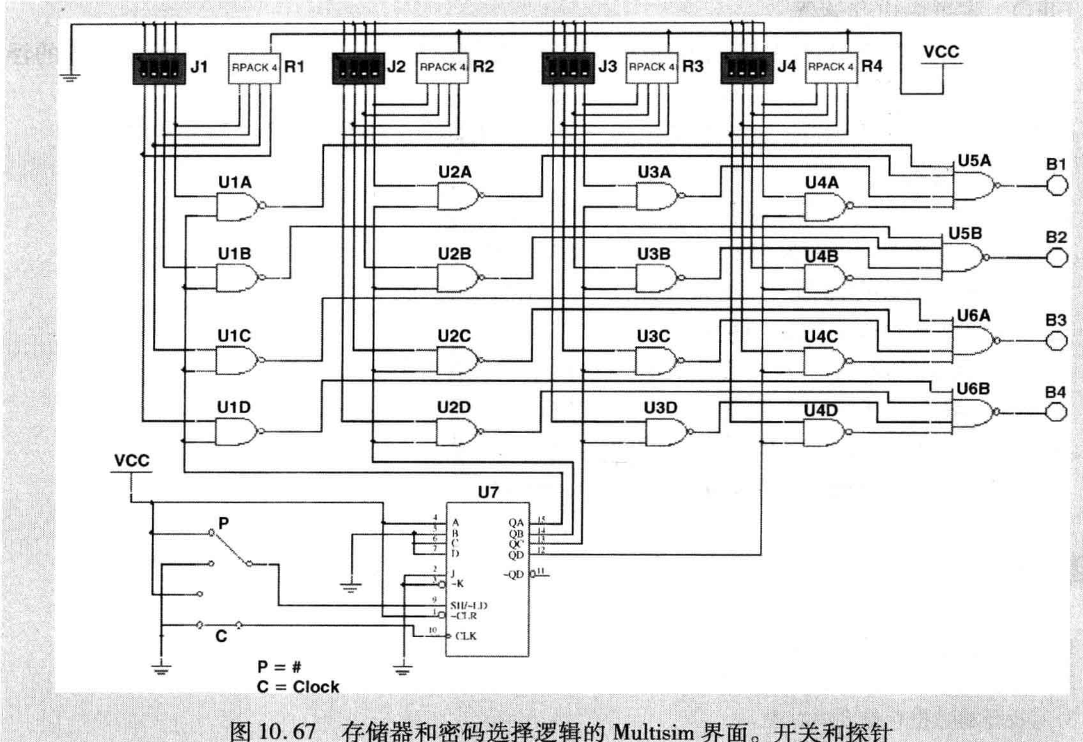


图 10.67　存储器和密码选择逻辑的 Multisim 界面。开关和探针灯仅为了测试的目的。当探针灯点亮时，表示输出1

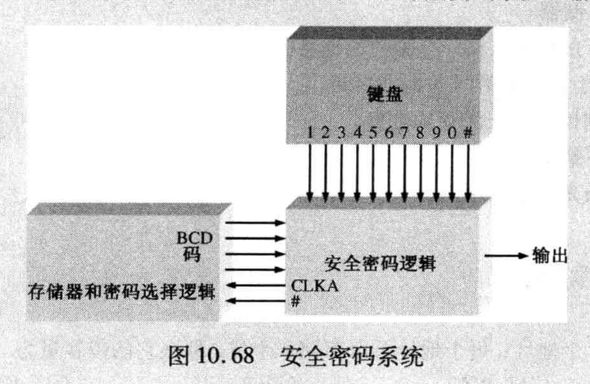


图 10.68　安全密码系统

关键词

地址　给出的存储器单元的位置或存储器中单元组的位置。

总线　基于一个标准化规格的一个或多个设备接口的相互连接。

字节　8 个位的一组数据。

容量　存储器可以存储的数据单位的总数目(位，字节，字)。

单元　存储器中的一个单一的存储元素。

DRAM　动态随机访问存储器；一组半导体存储器，使用电容作为存储单元，并且是易失的读/写存储器。

EPROM　可擦除、可编程只读存储器；半导体存储设备的一种，典型的使用紫外光线擦除数据。

FIFO 先进先出存储器。

闪存 非易失性读/写随机访问半导体存储器，数据以电荷形式存储在一种场效应管的浮门上。

硬盘 一种磁存储设备；通常为包装在一个封闭的盒子中的两个或多个磁片的堆叠。

LIFO 后进先出存储器；一种存储堆栈。

存储器 计算机的一部分或存储二进制数的其他系统。

PROM 可编程只读存储器；半导体存储器的一种。

RAM 随机访问存储器；一种易失性读/写半导体存储器。

读 从存储器中取回数据的过程。

ROM 只读存储器；一种非易失性随机访问半导体存储器。

SRAM 静态随机访问存储器；一种易失性读/写半导体存储器。

字 一组位或字节，可以作为单个实体存储在存储器中的一个位置；两个字节。

写 在存储器中存储数据的过程。

判断题 （答案在本章的结尾。）

1. 一个数据字节由 8 位组成。
2. 一个存储单元可以存储一个数据字节。
3. 写操作将数据存储到内存中。
4. 读操作总是擦除数据字节。
5. RAM 是随机地址存储器。
6. 如果一个静态 RAM 的电源关闭，存储的数据将丢失。
7. 高速缓冲存储器用于直接或暂时的数据存储。
8. 动态 RAM 必须周期性地刷新保存的数据。
9. ROM 是随机输出存储器。
10. 一个闪存使用一条光束存储数据。

自测题 （答案在本章的结尾。）

1. 一个存储器有 1024 个地址，每个地址可以存储 8 个位，那么它的位容量为
   (a) 1024 (b) 8192 (c) 8 (d) 4096
2. 一个 32 位数据字由下面的哪一项组成？
   (a)2 字节 (b)4 个半字节 (c)4 个字节 (d)3 个字节和 1 个半字节
3. 数据在什么时候存储在随机访问存储器(RAM)中？
   (a)读操作 (b)使能操作 (c)写操作 (d)寻址操作
4. 随机访问存储器(RAM)中给定地址中存储的数据在什么情况下会丢失？
   (a)电源关闭 (b)数据从地址中读出
   (c)在地址中写入新数据 (d)答案(a)和(c)
5. 一个 ROM 是
   (a)非易失性存储器 (b)易失性存储器 (c)读/写存储器 (d)以字节组织的内存
6. 具有 256 个地址的存储器有
   (a)256 条地址线 (b)6 条地址线 (c)1 条地址线 (d)8 条地址线
7. 以字节组织的存储器有

(a)1 条数据输出线　(b)4 条数据输出线　(c)8 条数据输出线　(d)16 条数据输出线

8. SRAM 中的存储单元是

(a)触发器　(b)电容器　(c)熔丝　(d)磁性区域

9. 一个 DRAM 必须

(a)定期置换　(b)定期刷新　(c)一直使能　(d)在每次使用之前编程

10. 闪存是

(a)易失的　(b)只读存储器　(c)读/写存储器　(d)非易失的

(e)答案(a)和(c)　(f)答案(c)和(d)

11. 硬盘、软盘、Zip 盘和 REV 盘都是

(a)磁-光存储设备　(b)半导体存储设备

(c)磁存储设备　(d)光存储设备

12. 光存储设备使用

(a)紫外线光　(b)电磁场　(c)光耦合　(d)激光

习题

10.1 节　半导体存储器基础

1. 识别图 10.69 中的 ROM 和 RAM。

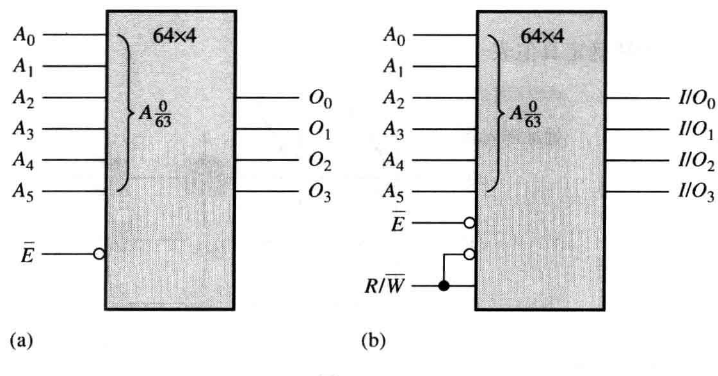


图 10.69

2. 解释 RAM 和 ROM 为什么都是随机存储器。

3. 解释地址总线和数据总线的用途。

4. 下面每个十六进制数分别表示哪个存储地址(0～256)：

(a)$0A_{16}$　(b)$3F_{16}$　(c)CD_{16}

10.2 节　随机访问存储器(RAM)

5. 一个静态存储器阵列具有 4 行，并且类似于图 10.9 中的存储器，其初始时存储的都是 0。在下面的条件之后，它的内容是什么？假设一个 1 选择一行。

行 0 = 1，数据进入(位 0) = 1

行 1 = 0，数据进入(位 1) = 1

行 2 = 1，数据进入(位 2) = 1

行 3 = 0，数据进入(位 3) = 0

6. 为 512 × 8 位的静态 RAM 绘制基本逻辑框图，给出所有的输入和输出。

7. 假设某个 64 k × 8 SRAM 的结构类似于图 10.11 中的 SRAM，确定它的存储单元阵列中的行和 8 位列的数目。

8. 为一个 64 k×8 存储器重新绘制图 10.11 中的模块图。

9. 解释 SRAM 和 DRAM 之间的区别。

10. 具有 12 条地址线的 DRAM，其容量是多少？

10.3 节 只读存储器(ROM)

11. 对于图 10.70 中的 ROM 阵列，确定所有可能输入组合的输出，并以表格的形式把它们总结出来(深色单元为 1，浅色单元为 0)。

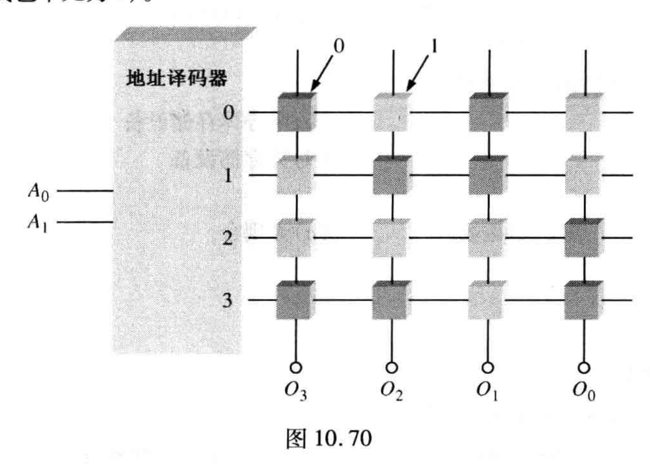


图 10.70

12. 为图 10.71 中的 ROM 确定真值表。

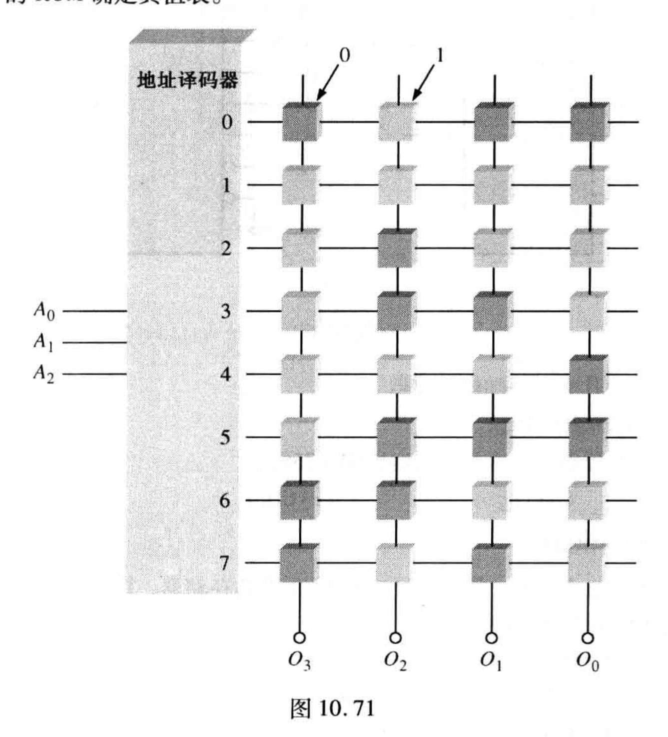


图 10.71

13. 使用类似于例 10.1 的过程，设计一个 ROM 用以将单个数字 BCD 变换为余 3 码。

14. 某个 ROM 具有 14 条地址线和 8 个数据输出，那么其总的位容量是多少？

10.6 节 存储器扩展

15. 使用 16 k×4 的 DRAM 来构建一个 64k×8 的 DRAM。画出逻辑框图。

16. 使用框图，给出怎样扩展 64k×1 的动态 RAM，以构建一个 256k×4 的 RAM。

17. 习题 15 和习题 16 中存储器的字长和字容量是多少？

10.7 节　特殊类型的存储器

18. 完成图 10.72 中的时序图，画出初始为 0 的 FIFO 串行存储器的输出波形，和图 10.49 中所给出的存储器的输出波形相似。

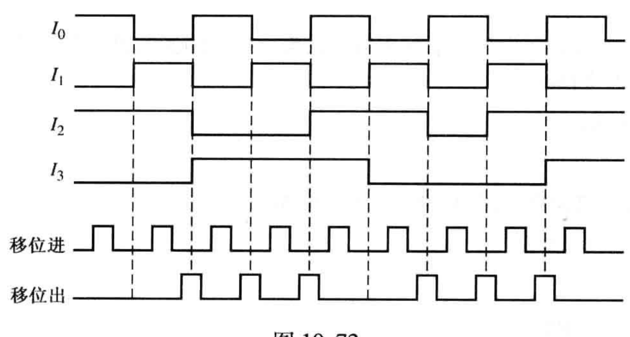


图 10.72

19. 考虑一个 4096×8 RAM，其中最后 64 个地址用做 LIFO 堆栈。如果 RAM 中的第一个地址是 000_{16}，指出用做堆栈的这 64 个地址。

20. 在习题 19 的存储器中，16 个字节被推入堆栈中。第一个字节所处的地址是什么？最后一个字节所处的地址是什么？

10.8 节　磁和光存储

21. 描述硬盘通常的格式化。

22. 解释硬盘驱动器的寻道时间和传输延迟周期。

23. 为什么磁带比磁盘需要更多的访问时间？

24. 解释磁-光盘、CD-ROM 和 WORM 的区别。

数字系统应用

25. 为图 10.66 的密码选择逻辑设计一个时序图，说明输入数字 4321 的序列。

26. 选择两个密码之后，图 10.66 的移位寄存器的状态是什么？

27. 在系统应用中开关的用途是什么？

28. 讨论在仿真时使用机械开关作为存储器组合的优点。

特殊设计问题

29. 给出可以用于实现图 10.67 逻辑框图的逻辑电路。

30. 给出 5 位安全密码的存储器和密码选择逻辑的逻辑框图。

答案

温故而知新

10.1 节　半导体存储器基础

1. 位是最小的数据单位。

2. 256 字节是 2048 位。

3. 一个写操作在存储器中存入数据。

4. 一个读操作从存储器中获取数据。

5. 一个数据单位由它的地址定位。

6. RAM 是易失性的，有读/写能力。一个 ROM 是非易失性的，仅有读数的能力。

10.2 节 随机访问存储器(RAM)

1. 具有突发特性的异步和同步。
2. 在计算机和主存储器之间的一个小的速度较快的存储器。
3. SRAM 具有锁存单元，只要加电源，就可以一直保留数据。DRAM 具有电容存储单元，必须定期刷新。
4. 由于电容的放电，所以刷新操作用于防止数据丢失。通过刷新电容，使得存储的位恢复应有的电平。
5. FPM，EDO，BEDO，同步。

10.3 节 只读存储器(ROM)

1. 512 × 8 等于 4069 位。
2. 掩模 ROM，PROM，EPROM，UV EPROM，EEPROM。
3. 256 字节的位置需要 8 位地址。

10.4 节 可编程 ROM

1. PROM 是场可编程；ROM 不是。
2. EPROM 擦除以后剩下的是 1。
3. 读是 PROM 的正常操作。

10.5 节 闪存

1. 闪存、ROM、EPROM 和 EEPROM 是非易失性的。
2. 闪存是非易失性的，SRAM 和 DRAM 是易失性的。
3. 编程，读，擦除。

10.6 节 存储器扩展

1. 8 个 RAM。
2. 8 个 RAM。
3. SIMM；单顺序存储模式。
4. DIMM：双顺序存储模式。
5. RIMM：内存总线内联存储模式。

10.7 节 特殊类型的存储器

1. 在 FIFO 存储器中，第一个进入的位是第一个出去的位。
2. 在 LIFO 存储器中，最后一个进入的位是第一个出去的位。
3. 入栈操作或入栈指令把数据加到存储器堆栈中。
4. 出栈操作或出栈指令把数据从存储器中取出。
5. CCD 是电荷耦合设备。

10.8 节 磁和光存储

1. 磁性存储器：软盘，硬盘，磁带，磁-光盘。
2. REV 存储容量是 35 GB 或 70 GB。
3. 磁盘由磁道和扇区组成。
4. 磁-光盘使用一个激光束和一个电磁。
5. 光存储：CD-ROM，CD-R，CD-RW，DVD-ROM，WORM。

例题的相关问题

10.1 $G_3G_2G_1G_0 = 1110$。

10.2 并行连接 8 个 64k × 1 ROM，形成 64k × 8 ROM。

10.3　16 个 64k × 1 ROM。

10.4　参见图 10.73。

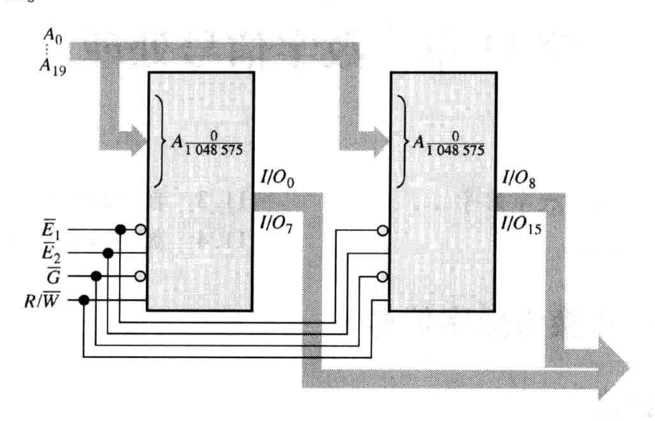


图 10.73

10.5　ROM 1:0 到 524 287；ROM 2：524 288 到 1 048 575。

判断题

1. T　2. F　3. T　4. F　5. F　6. T　7. T　8. T　9. F　10. F

自测题

1. (b)	2. (c)	3. (c)	4. (d)	5. (a)	6. (d)
7. (c)	8. (a)	9. (b)	10. (f)	11. (c)	12. (d)

第 11 章 数字信号处理

章节提纲

11.1 模拟信号转换为数字信号

11.1.1 采样和滤波

一个抗混叠滤波器和一个采样保持电路是数字信号处理系统中常见的两种电路。采样保持功能完成两个操作，第一是采样。采样就是在一个波形上获取足够数量的离散值，这些值将描述这个波形形状的特征。采样值越多，描述的波形就越准确。采样把模拟信号转换成一系列脉冲，每个脉冲表示信号在某个给定时刻的振幅。采样过程如图 11.1 所示。

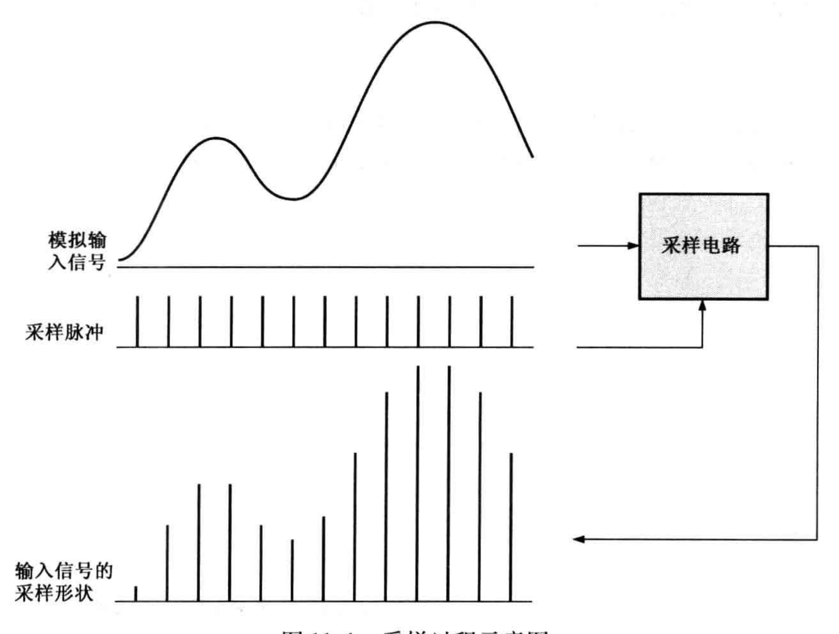


图 11.1 采样过程示意图

模拟信号被采样时必须满足一定的条件，以便准确地表达原始信号。所有的模拟信号(除了纯粹的正弦波)都包含一些频率组成的频谱。对于纯粹的正弦波，多种频率的组合称为谐波(harmonics)。模拟信号的谐波是一些不同频率和振幅的正弦波。

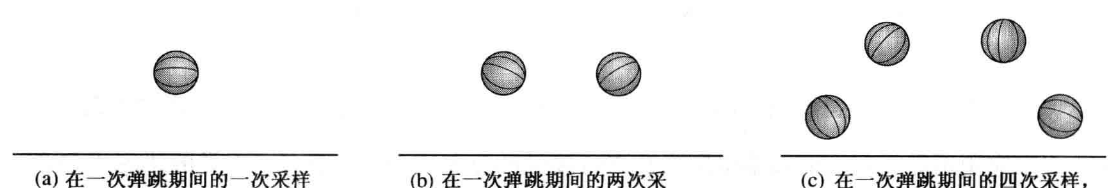
当一个给定周期波的谐波加在一起时，结果就得到原始信号。在对信号采样之前，必须经过一个低通滤波器(抗混叠滤波器)，以去除超过确定值的谐波频率，这个确定值取决于奈奎斯特频率。

采样定理 注意图 11.1 中有两个输入波形。一个是模拟信号，另一个是采样脉冲波形。采

样定理的陈述是，要表示一个模拟信号，采样频率f_{sample}必须至少是模拟信号最高的频率成分$f_{a(max)}$的两倍。另一种表达方式是，最高模拟频率不能高于采样频率的一半。频率$f_{a(max)}$称为“奈奎斯特频率”，由式(11.1)表示。在实际情况下，采样频率应当高于最高模拟频率的两倍。

$$f_{sample} > 2f_{a(max)} \quad (11.1)$$

为了直观地理解采样定理，一个简单的“弹跳球”的比喻很有用。尽管它不能十分完美地表达电子信号的采样过程，但它确实能够说明其基本想法。如果一个球在弹跳期间的一个瞬间被照相(采样)，如图 11.2(a)所示，这时不能给出任何有关这个球路径的情况，除了说明它离开地板之外，而不能判断这个球向上、向下或弹跳的高度。如果一次弹跳期间，在两个相等时间间隔的瞬间拍照，如图 11.2(b)所示，那么只能得到关于球运动情况的最少的信息量，而不知道球弹跳的距离。在这种特殊的情况下，仅知道在两次拍照的时间里球是在空中，球弹跳的最大高度比每张照片中的高度都高或至少相等。如果拍摄四张照片，如图 11.2(c)所示，那么球在一次弹跳期间所经过的路径开始呈现出来。拍的照片(采样)越多，可以确定的球的弹跳路径就越精确。



(a) 在一次弹跳期间的一次采样

(b) 在一次弹跳期间的两次采样，这是描述任何有关球的运动所需要的最低的要求

(c) 在一次弹跳期间的四次采样，形成球弹跳路径的粗略轨迹

图 11.2　采样定理的弹跳球模拟

滤波的需要　低通滤波对于消除模拟信号的谐波频率是十分必要的，谐波频率指所有超过奈奎斯特频率的那一部分。如果模拟信号中的任何谐波频率超过了奈奎斯特频率，就会出现不想得到情况，称之为发生假信号混叠。假信号是在采样频率小于信号频率的两倍时产生的信号。假信号的频率小于被采样模拟信号的最高频率，因此在输入模拟信号的频谱或频带范围内造成失真。这样的信号实际上会“假装”成为模拟信号的一部分，但实际情况并非如此，因此称为“假信号”。

另一种观察混叠假信号的方法是考虑采样脉冲产生一个高于和低于采样频率的谐波频谱，如图 11.3 所示。如果模拟信号包含高于奈奎斯特频率的频率，那么这些频率将与采样波形的频谱重叠，如图所示，从而产生干扰。采样波形中较低成分的频率和模拟波形的频谱混合，结果会产生一个假信号混叠的错误。

必须使用低通抗混叠滤波器，以便对一个给定的采样频率限制模拟信号的频谱。为了避免假信号混叠错误，滤波器至少必须把所有超过最小频率的模拟信号频率去除，这个最小频率就是采样频谱中的最小频率，如图 11.4 所示。还可通过把采样频率提高到足够大的方法来避免假信号混叠错误。但是，最高采样频率通常会受到其后的模-数转换器(ADC)性能的限制。

应用　采样的一个应用例子是在数字音频设备中，使用的采样速率为 32 kHz、44.1 kHz或48 kHz(每秒钟的采样次数)。48 kHz 的速率最常用，但 44.1 kHz 的速率常用于音频 CD 和预先录制好的磁带。根据奈奎斯特速率，采样频率至少必须是音频信号的两倍。因此，44.1 kHz 的 CD 采样速率可获取大约最高为 22 kHz 的频率，这个频率超过了多数音频设备常用的 20 kHz 的规定频率。

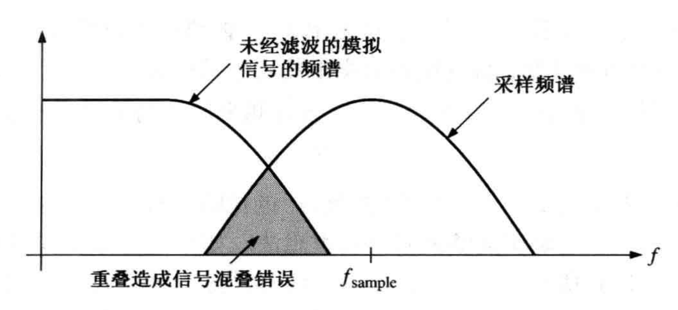


图 11.3 $f_{sample} < 2f_{a(max)}$ 情况的基本解释

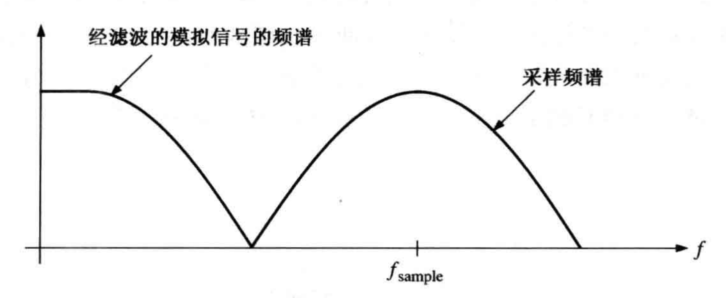


图 11.4 经过低通滤波，模拟信号和采样信号的频谱不会重叠，从而消除了假信号混叠错误

许多应用不需要很宽的频率范围来获取可接受的复制的声音。例如，人类的语音中包含一些接近 10 kHz 的频率信号，因此，需要采样速率至少是 20 kHz。但是，如果仅复制 4 kHz 以内的频率(理想情况下，需要最小 8 kHz 的采样速率)，复制出的语音就可以听得很清楚。另一方面，如果没有以一个足够高的速率对声音信号采样，那么假信号混叠的效果就会以背景噪声和失真的形式变得很明显。

11.1.2 保持采样值

保持操作是采样保持功能的第二部分。经过滤波和采样，取得的值必须保持不变，直到下一次采样的发生。这是必须进行的操作，以便让模-数转换器有时间去处理采样值。采样和保持操作的结果就是得到近似模拟输入信号波形的“楼梯”形状的波形，如图 11.5所示。

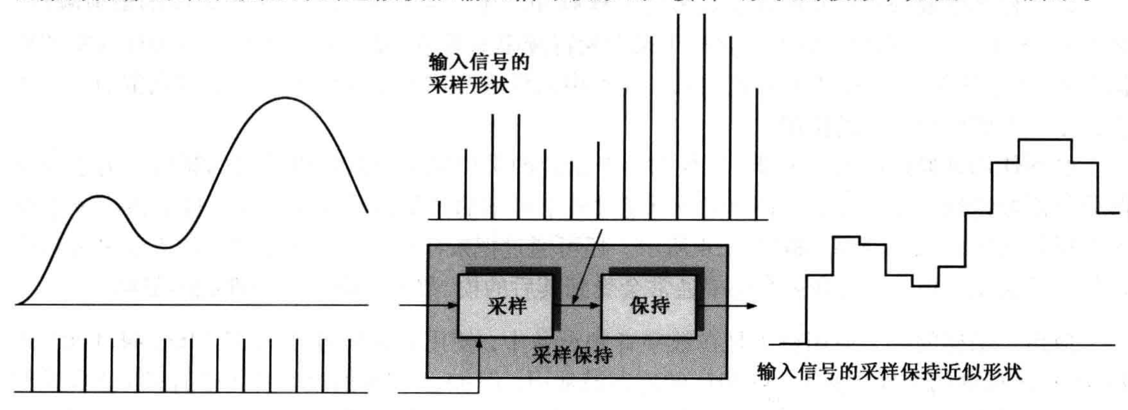


图 11.5 采样保持操作示意图

11.1.3　模-数转换器

模-数转换是把采样保持电路的输出转换成一系列二进制数的过程，这些二进制数表示在每个采样时刻模拟输入的振幅。采样保持过程将保持模拟输入信号在两次采样脉冲之间的幅度不变；因此，在转换间隔期间，即采样脉冲之间的时间内，模-数转换可以使用一个常数值作为转换结果，而不会有模拟信号那样的变化。图 11.6 给出了模-数转换器(ADC)的基本功能。采样时间间隔由虚线指示。

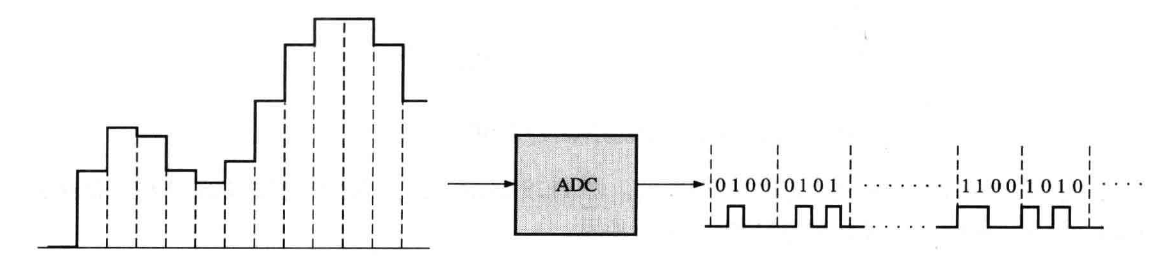


图 11.6　模-数转换器的基本功能(二进制数和位数仅仅是为了例证而任意选择的)。表示ADC输出波形的二进制数也显示在图中

量化　将模拟值转换为二进制数的过程称为量化。在量化过程中，ADC 把模拟信号的每个采样值都转换为一个二进制数。用来表示采样值的位数越多，表示的精度就越高。

为了说明，把模拟信号波形的再现量化为 4 个级别(0～3)。如图 11.7 所示，需要两位码，在垂直坐标轴上每个量化级别由一个两位码表示，沿着水平坐标轴的每个采样间隔都有编号。采样得到的数据在整个采样周期里一直保持。数据量化为下一个低电平，如表 11.1 所示(例如，比较采样点 3 和 4，赋予不同的电平)。

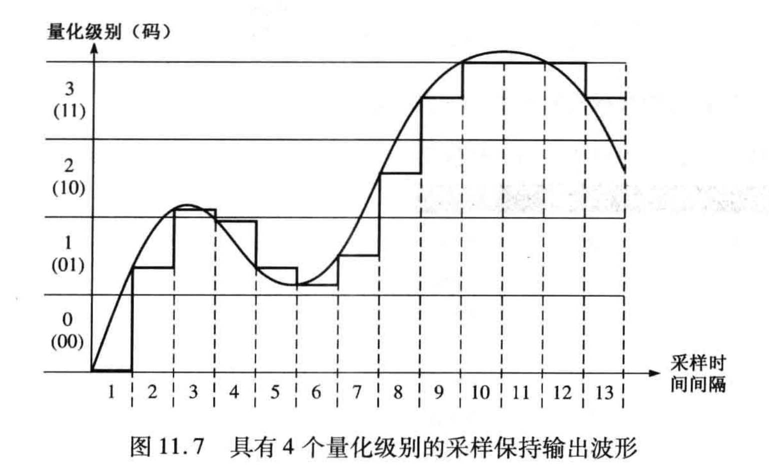


图 11.7　具有 4 个量化级别的采样保持输出波形

表 11.1　图 11.7 所示波形的两位量化码

采样时间间隔	量化级别	码
1	0	00
2	1	01
3	2	10
4	1	01
5	1	01
6	1	01
7	1	01
8	2	10
9	3	11
10	3	11
11	3	11
12	3	11
13	3	11

如果使用得到的 2 位码重构原始波形[这由数-模转换器(DAC)实现]，可以得到如图 11.8 所示的波形。可见，仅使用两位码表示采样值会使精度降低很多。

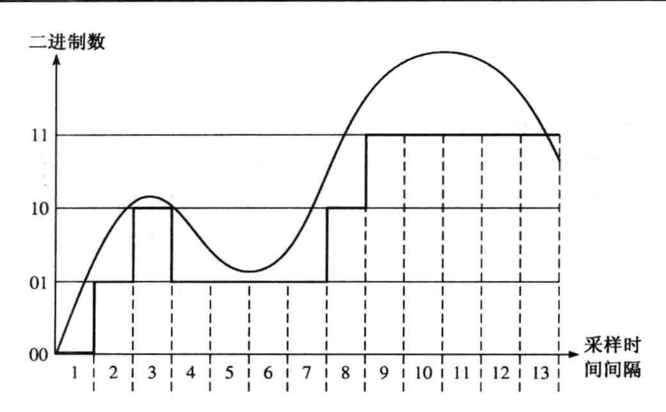


图 11.8 使用 4 个量化级别(2 位)重构图 11.7 中的波形

现在来看更多位的码是如何提高精度的。图 11.9 显示了使用 16 个量化级别(4 位)来表示的同一波形。4 位量化过程的总结如表 11.2 所示。

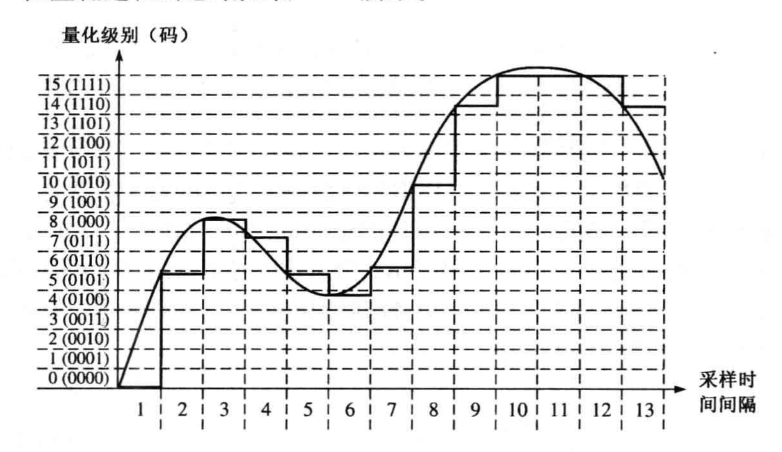


图 11.9 具有 16 个量化级别的采样保持输出波形

表 11.2 图 11.9 中波形的 4 位量化

采样时间间隔	量化级别	码
1	0	0000
2	5	0101
3	8	1000
4	7	0111
5	5	0101
6	4	0100
7	6	0110
8	10	1010
9	14	1110
10	15	1111
11	15	1111
12	15	1111
13	14	1110

如果使用得到的 4 位码重构原始波形，就可以得到如图 11.10 所示的波形。比起在图 11.8 中使用 4 个量化级别的情况，这次得到的结果更接近于原始波形。这表明使用更多的量化位，可以得到更高的精度。多数集成电路 ADC 使用 8 位到 24 位，采样保持功能有时就包含在 ADC 芯片中。在接下来的小节中将介绍几种类型的 ADC。

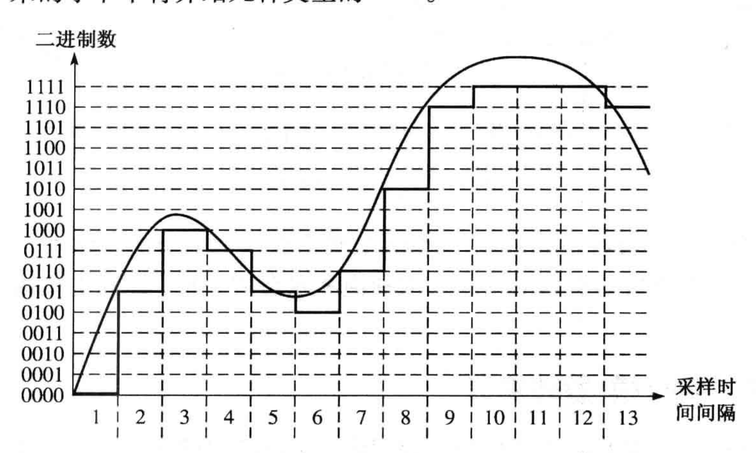


图 11.10　使用 16 个量化级别(4 位)重构图 11.9 中的波形

11.1 节　温故而知新

1. 采样是什么意思?
2. 为什么必须保持采样值?
3. 如果模拟信号的最高频率是 20 kHz，最小的采样频率是多少?
4. 量化是什么意思?
5. 量化处理的精度由什么确定?

11.2　模-数转换方法

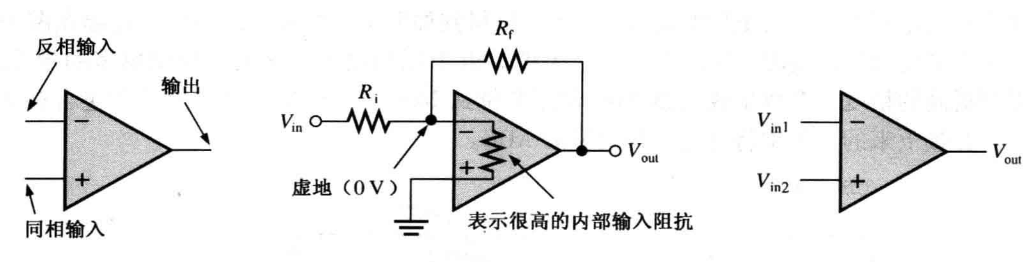
11.2.1　快速浏览运算放大器

在开始介绍模-数转换器(ADC)之前，先简单看看一种元件，它对于多数类型的 ADC 和数-模转换器(DAC)都是常用的元件。这种元件就是运算放大器，简称 op-amp，即运算放大器的缩写形式。

运算放大器是一种线性放大器，它有两个输入(反相和同相)和一个输出。它有一个很高的电压增益和很大的输入阻抗，以及很低的输出阻抗。运算放大器的符号如图 11.11(a)所示。如果用做反相放大器时，那么运算放大器的连接如图 11.11(b)所示。根据式(11.2)，反馈电阻 R_f 和输入电阻 R_i 控制电压增益，其中 V_{out}/V_{in} 是闭环电压增益(闭环是指由电阻 R_f 提供的从输出到输入的反馈)，负号表示反相。

$$\frac{V_{out}}{V_{in}} = -\frac{R_f}{R_i} \tag{11.2}$$

在反相放大器的连接中，运算放大器的反相输入接近地电位(0 V)，因为反馈和极高的开环增益使得两个输入之间的差分电压非常小。由于同相输入接地，反相输入接近 0 V，所以称为虚地。



(a) 运算放大器的符号　(b) 运算放大器用做反相放大器，增益为 R_f/R_i　(c) 运算放大器用做比较器

图 11.11　运算放大器

把运算放大器用做比较器如图 11.1(c)所示，其中有两个电压加到输入。当两个输入电压有很小的差别时，运算放大器的输出就进入两种饱和状态的一种，即高电平或低电平，这取决于哪一个输入电压大。

11.2.2　快速(同时)模-数转换器

快速方法利用高速比较器在参考电压和模拟输入电压之间进行比较。如果给定比较器的输入电压超过参考电压，就产生高电平。图 11.12 给出了一个 3 位转换器，使用 7 个比较器的电路；对于全部为 0 的情况不需要比较器。这种类型的 4 位转换器需要 15 个比较器。一般情况下，对于 n 位二进制码的转换，需要 2^n-1 个比较器。ADC 所用的位数就是它的分辨率。对于较多位数的二进制数需要大量的比较器，这是快速 ADC 的缺点之一。它的主要优点是提供了一个快速转换时间，由于它的一个高“通过量”，因此以每秒采样次数(sps)来衡量。

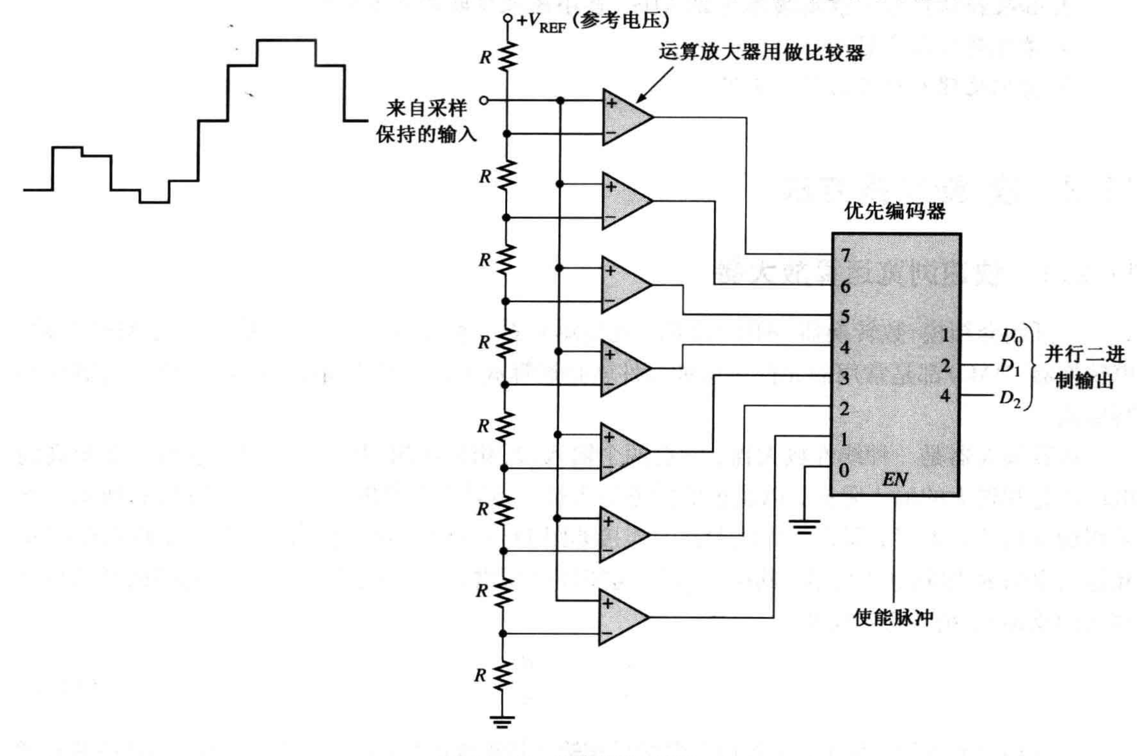


图 11.12　3 位快速 ADC

每个比较器的参考电压都由电阻分压电路设定。每个比较器的输出都连接到优先权编码器的一个输入。编码器由 *EN* 输入上的一个脉冲使能，3 位码表示出现在编码器输出上的输入值。二进制数由最高级别输入的高电平决定。

二进制数的序列表示 ADC 的输入，转换的精度由使能脉冲的频率和二进制编码的位数确定。每次使能脉冲有效的期间，输入信号得到采样。

例 11.1　对于如图 11.13 的输入信号和编码器使能脉冲，确定图 11.12 中的 3 位快速 ADC 的二进制编码输出，其中 $V_{REF}=8\ V$。

解： 结果得到的数字输出序列如下所示，并给出了和使能脉冲相关的图 11.14 所示的波形图：

100，110，111，110，100，010，000，001，011，101，110，111

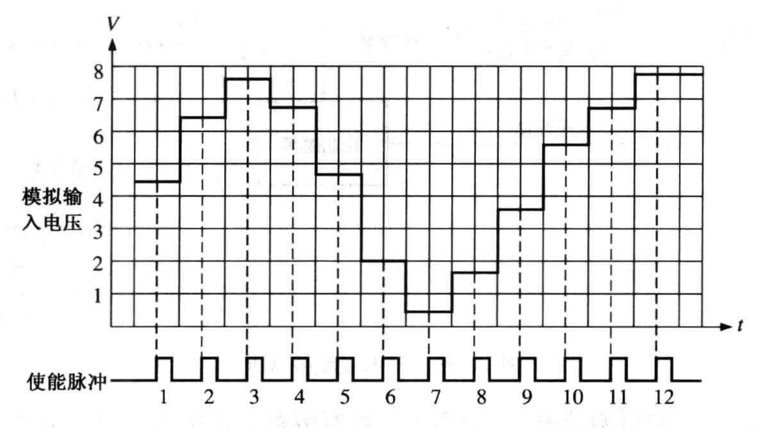


图 11.13　把一个波形转换为二进制数的采样值

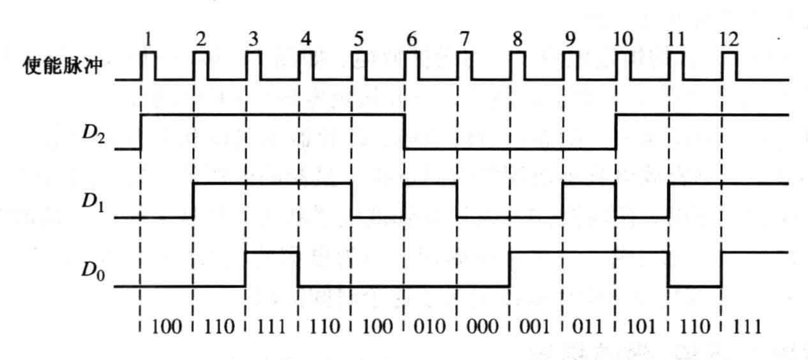


图 11.14　所得到的采样保持值的数字输出。输出 D_0 是 3 位二进制数的最低有效位

相关问题[①]：如果把图 11.13 中的使能脉冲的频率减半，确定由 6 个脉冲得到的数字输出序列所表示的二进制数。其中是否丢失了信息？

11.2.3　双积分模-数转换器

双积分 ADC 在数字电压表和其他类型的测量仪器中很常见。一个斜坡发生器（积分器）用来产生双积分特性。双积分 ADC 的框图如图 11.15 所示。

图 11.16 解释了双积分的转换。一开始假设计数器复位，积分器的输出为零。这时假设正

① 相关问题的答案请参见本章末尾。

的输入电压 V_{in} 通过开关(SW)加在积分器的输入，这由控制逻辑选择。因为运算放大器 A_1 的反相输入端处在虚地状态，并且假设输入电压 V_{in} 在一段时间是常数，这时通过输入电阻 R 的电流为常数，因此通过电容 C 的电流也是常数。因为电流是常数，所以电容线性充电，结果在运算放大器 A_1 的输出有一个向下的线性电压斜坡，如图 11.16(a)所示。

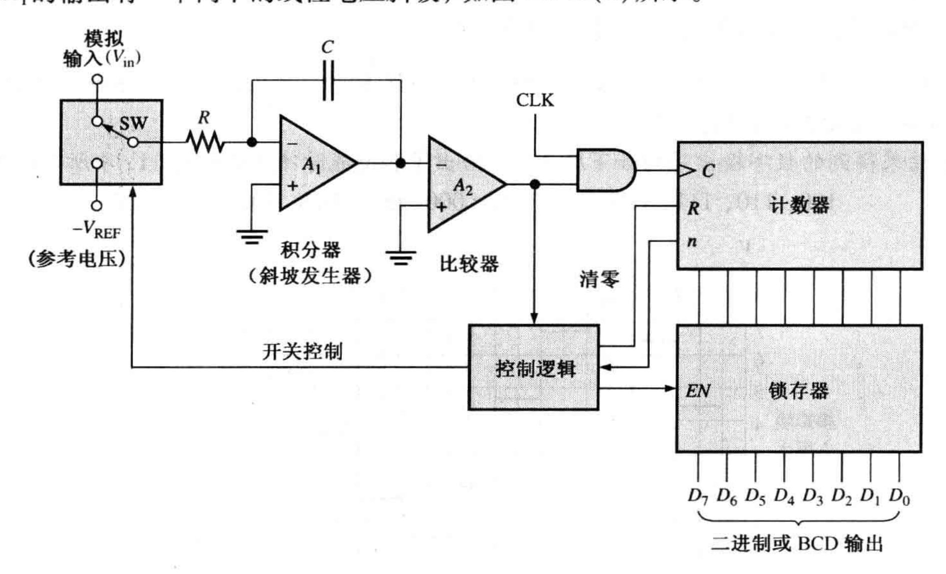


图 11.15 基本双积分 ADC

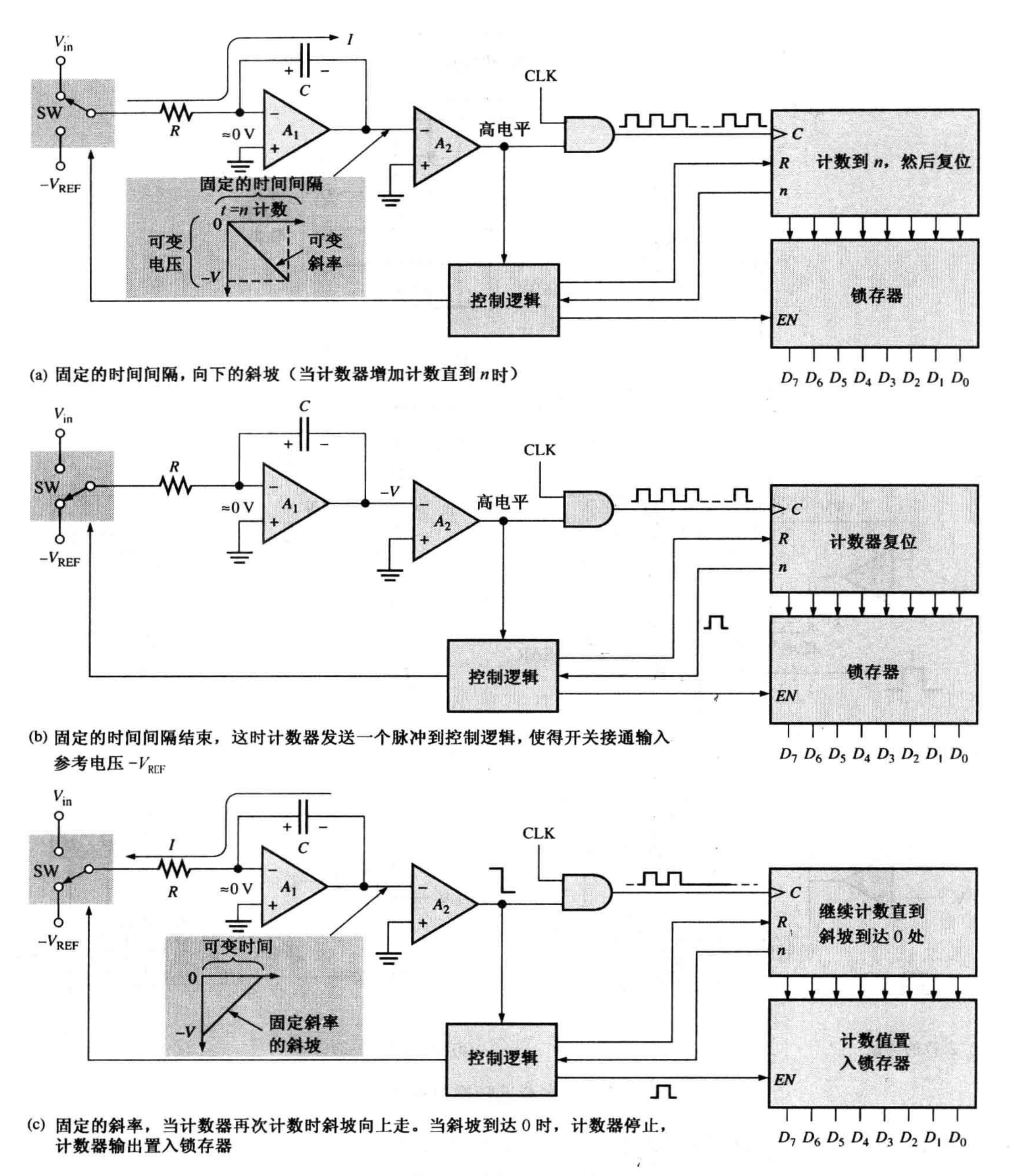
当计数器到达设定的计数值时，它就复位，控制逻辑电路把负的参考电压 $-V_{REF}$ 切换到运算放大器 A_1 的输入，如图 11.16(b)所示。在这一个时间点上，电容充电到达负电压($-V$)，这个负电压与输入模拟电压成比例。

这时由于来自 $-V_{REF}$ 的恒定电流，电容线性放电，如图 11.16(c)所示。这个线性放电过程在运算放大器 A_1 的输出产生一个正向斜坡，斜坡从负电压($-V$)开始，有一个恒定的斜率，这个斜率与前面的充电电压无关。随着电容的放电，计数器从复位状态开始计数。由于放电的速率(斜率)是不变的，电容放电到零所用的时间取决于最初的电压值 $-V$(正比于 V_{in})。当积分器(A_1)的输出电压到达零时，比较器(A_2)就切换到低电平状态，并且关闭计数器的时钟。这时二进制计数值被锁存，从而完成了一次转换周期。因为电容放电所用的时间仅仅和 $-V$ 有关，所以二进制计数值与 V_{in} 成比例，而计数器记录了这个时间间隔。

11.2.4 逐次渐近模-数转换器

模-数转换最广泛的使用方法之一是逐次渐近。这种方法的转换时间比双积分转换时间要少，但比快速方法要多。这种方法有一个固定的转换时间，对于任何模拟输入值，转换的时间相同。

图 11.17 给出了 4 位逐次渐近 ADC 的基本框图。它由 DAC(DAC 将在 11.3 节中介绍)、一个逐次渐近寄存器(SAR)和一个比较器组成。基本操作如下：从 DAC 输入位的最高有效位(MSB)开始，DAC 的输入位每次启用一个(使它等于 1)。随着每一位的启用，比较器就产生一个输出，指出输入信号电压是大于还是小于 DAC 的输出。如果 DAC 的输出大于输入信号，那么比较器的输出是低电平，导致寄存器中相应的位复位。如果 DAC 的输出小于输入信号，就会在寄存器中相应的位保留 1。这个系统遵循最高有效位优先的原则，然后是下一个最高有效位，接着是下一个，以此类推。在 DAC 的所有位都经过尝试后，这个转换周期就结束了。



(a) 固定的时间间隔，向下的斜坡（当计数器增加计数直到 n 时）

(b) 固定的时间间隔结束，这时计数器发送一个脉冲到控制逻辑，使得开关接通输入参考电压 $-V_{REF}$

(c) 固定的斜率，当计数器再次计数时斜坡向上走。当斜坡到达 0 时，计数器停止，计数器输出置入锁存器

图 11.16　双积分转换的解释

为了更好地理解逐次渐近 ADC 的操作过程，以一个特定的 4 位转换过程为例。图 11.18 给出了一个恒定输入电压(这里为 5.1 V)的转换步骤。假设 DAC 具有下列输出特性：对于 2^3 位(MSB)，$V_{out}=8$ V；对于 2^2 位，$V_{out}=4$ V；对于 2^1 位，$V_{out}=2$ V；对于 2^0 位(LSB)，$V_{out}=1$ V。

图 11.18(a)给出了转换周期的第一步，MSB = 1。DAC 的输出是 8 V。因为这个输出电压大于 5.1 V，所以比较器的输出是低电平，使得 SAR 中的最高有效位复位(0)。

图 11.18(b)给出了转换周期的第二步，2^2 位等于 1。DAC 的输出是 4 V。这个输出电压小于 5.1 V，所以比较器的输出转向高电平，使得 SAR 中的这位保持为 1。

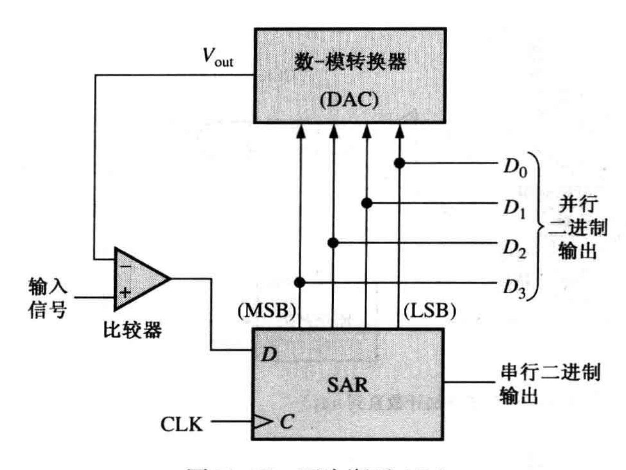


图 11.17 逐次渐近 ADC

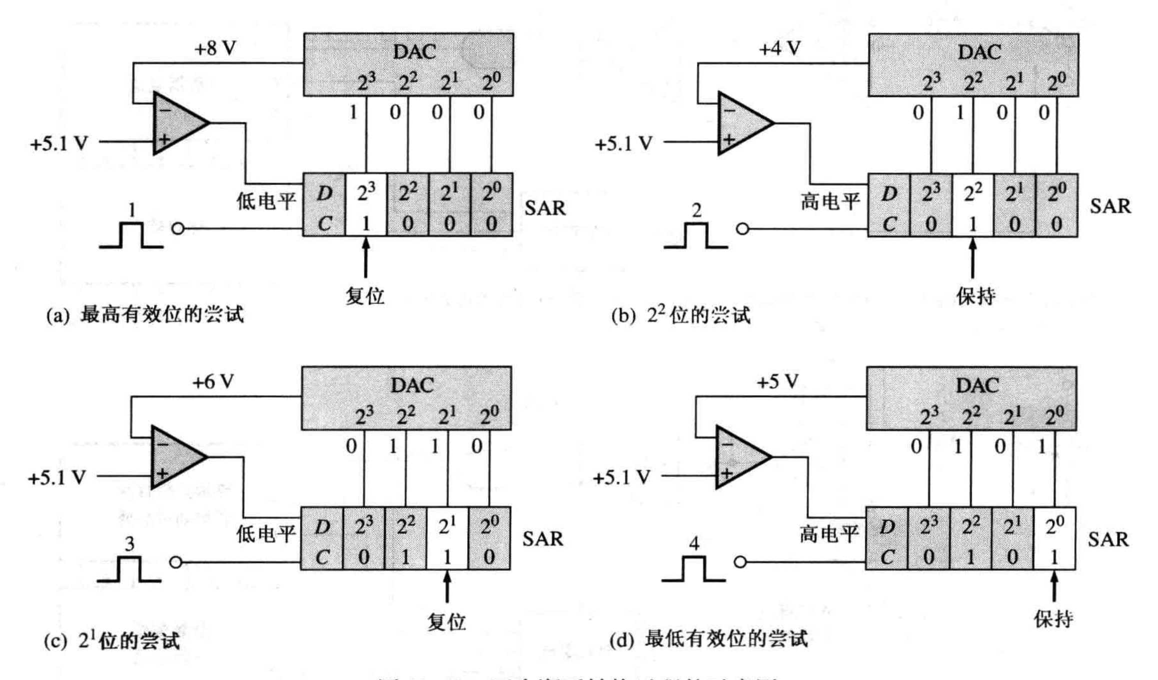


图 11.18 逐次渐近转换过程的示意图

图 11.18(c)给出了转换周期的第三步，2^1位等于 1。因为 2^2 位输入和 2^1 位输入是 1，4 V + 2 V = 6 V，所以 DAC 的输出是 6 V。这个输出电压大于 5.1 V，所以比较器的输出转向低电平，使得 SAR 中的这位复位(0)。

图 11.18(d)给出了转换周期的第四步也是最后一步，2^0位等于 1。因为在 2^2位输入和 2^0位输入都是 1，4 V + 1 V = 5 V，所以 DAC 的输出是 5 V。

这 4 位都经过了尝试，所以转换过程结束。此时，寄存器中的二进制编码为 0101，这个值接近输入 5.1 V 的二进制数。增加位数将得到更精确的结果。这时又开始另一个转换周期，这个基本过程又将重复。在转换周期的开始，SAR 清零。



ADC0804 模-数转换器

ADC0804 是逐次渐近 ADC 的一个例子。它的框图如图 11.19 所示。这个芯片的工作电压为 +5 V，8 位分辨率，转换时间为 100 μs。另外，芯片有一个自带的时钟发生器。作为选择，可以使用外部时钟。其中为三态数据输出，因此可以和微处理器的总线系统接口。

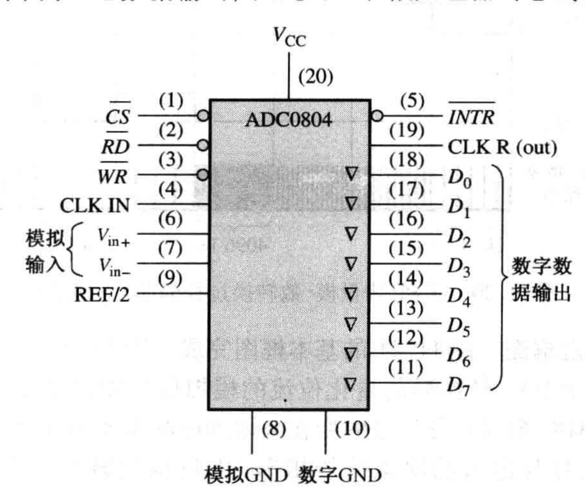


图 11.19　ADC0804 模-数转换器

这种芯片的基本操作如下：ADC0804 包含一个 256 电阻 DAC 网络的等效电路。逐次渐近逻辑对网络电路进行排序，以便使模拟差模输入电压差（$V_{in+} - V_{in-}$）与电阻网络的输出匹配。首先测试最高有效位。8 次比较后(64 个时钟周期)，8 位二进制数就传送到输出锁存器中，中断($\overline{INTR}$) 输出变为低电平。通过把 $\overline{INTR}$ 输出连接到写($\overline{WR}$)输入，并使转换开始($\overline{CS}$)保持为低电平，芯片可以工作在自由运行模式。为了确保在所有条件下都可启动，在通电周期，需要 $\overline{WR}$ 输入为低电平。任何时刻使 $\overline{CS}$ 为低电平随后都会中断转换过程。

当 $\overline{WR}$ 输入变为低电平时，内部逐次渐近寄存器和 8 位移位寄存器都会复位。只要 $\overline{CS}$ 和 $\overline{WR}$ 保持低电平，ADC 就保持在复位状态。在 $\overline{CS}$ 或 $\overline{WR}$ 从低电平变到高电平后的 1 ~ 8 个时钟周期，模-数转换开始。

当 $\overline{CS}$ 和 $\overline{RD}$ 输入都是低电平时，三态输出锁存器使能，并且输出码加到 $D_0 \sim D_7$ 线上。当 $\overline{CS}$ 或 $\overline{RD}$ 输入中的一个回到高电平，就会禁用 $D_0 \sim D_7$ 输出。

11.2.5　求和-增量模-数转换器

求和-增量(sigma-delta)是在模-数转换过程中广泛使用的转换方法，特别是在使用音频信号的通信领域。这种方法基于增量调制，即对两次连续采样间的差值(增加或减少)进行量化；另一种 ADC 方法是基于采样的绝对值。增量调制是一种 1 的位量化方法。

增量调制器的输出是一个单一的位数据流，其中相关的 1 和 0 的数目表示输入信号的电平或幅度。1 的数目超过给定时钟周期数，超出的数目确定了在这个时间间隔期间的振幅。1 的数目的最大值对应于输入正电压的最大值。1 的数目的一半对应于输入电压为零。没有 1 时(全 0)对应于负电压的最大值。图 11.20 以简化方式说明了这一点。例如，假设输入信号为正

的最大值的时间隔期间出现了 4096 个 1。因为零是输入信号动态范围内的中间点，所以当输入信号为零时，间隔期间会出现 2048 个 1。当输入信号为负的最大值时，那么在间隔期间没有 1。对于两者之间的信号电平，1 的数目与电平值成正比。

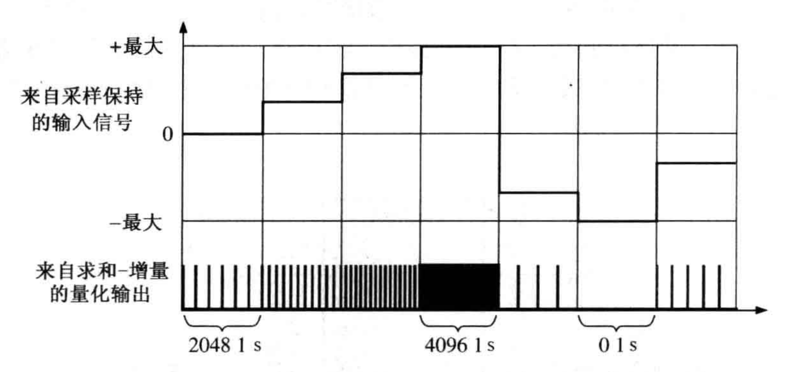


图 11.20　求和-增量模-数转换过程的简化示意图

求和-增量 ADC 功能框图　图 11.21 的基本框图完成了图 11.20 所示的转换过程。模拟输入信号和来自反馈回路中 DAC 转换后的量化位流的模拟信号加到求和(Σ)点。Σ 右边的差分(Δ)信号被积分，1 位 ADC 根据差分信号的变化来增加或减少 1 的数目。这样做的目的是为了保持反馈回来的量化信号与进入的模拟信号相等。1 位量化器本质上就是比较器后面跟锁存器。

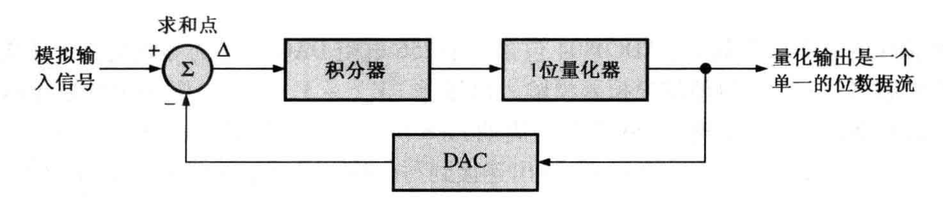


图 11.21　求和-增量 ADC 的部分功能框图

为了使用一种特殊方法完成求和-增量转换过程，将一个单一的位数据流转换成为一系列二进制编码，如图 11.22 所示。计数器对连续间隔的量化数据流中的 1 进行计数。这时计数器中的码表示每个时间间隔的模拟输入信号的幅度。这些码移位进入锁存器暂时保存。锁存器的输出是一系列的 n 位码，它们完整地表示这个模拟信号。

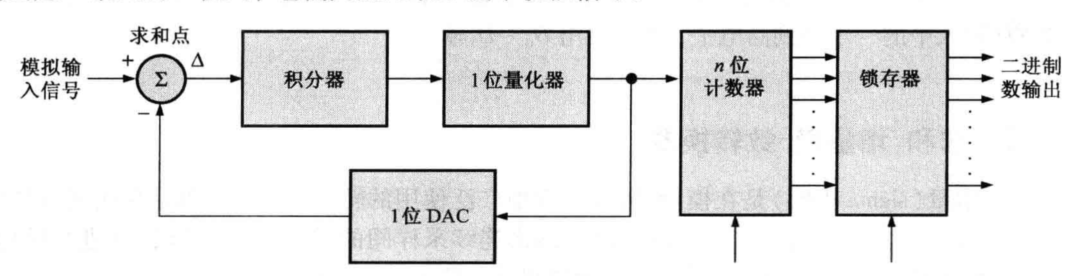


图 11.22　一种类型的求和-增量 ADC

11.2.6　模-数转换器的测试

测试 ADC 的一种方法如图 11.23 所示。一个 DAC 用做测试设备的一部分，它把 ADC 的输出转换回模拟形式，用来与测试输入信号进行比较。

测试输入信号以线性斜坡的形式施加在 ADC 的输入上。然后把得到的二进制输出序列加在 DAC 测试单元，并转换为楼梯状的斜坡。输入和输出的斜坡进行比较以确定是否有偏差。

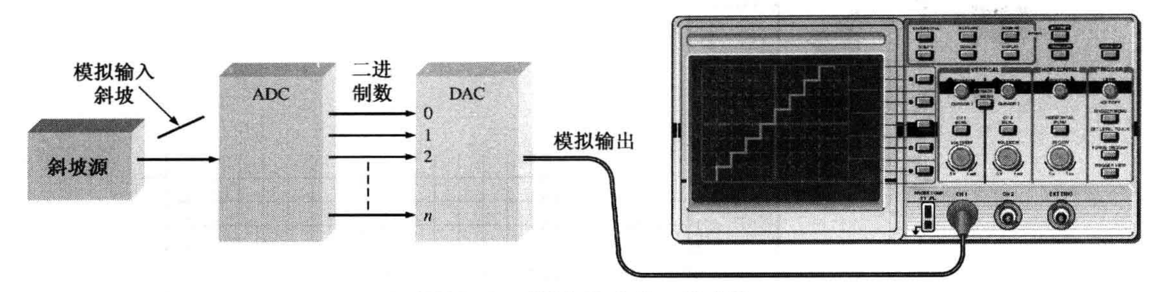


图 11.23　测试 ADC 的一种方法

11.2.7　模-数转换错误

再次使用 4 位转换过程来说明这些原则。假定测试输入信号是一个理想的线性斜坡。

丢失的码　如图 11.24(a)所示的楼梯状输出，它表示二进制数 1001 没有出现在 ADC 的输出。注意，码值 1000 保持了两个时间间隔，然后输出跳到 1010。

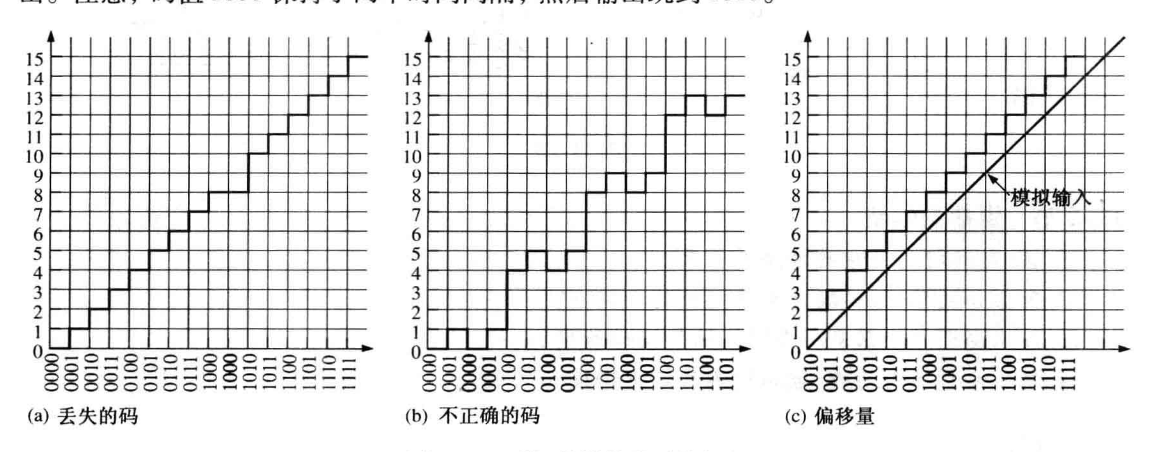


图 11.24　模-数转换错误的解释

例如，在快速 ADC 中，运算放大器比较器的一次失败操作可能导致丢失码的错误。

不正确的码　如图 11.26(b)所示的楼梯状输出，它表示有几个从 ADC 出来的二进制码字是不正确的。经过分析表明，在这种特殊情况下，2^1位线停留在低电平(0)状态。

偏移量　偏移量的情况如图 11.24(c)所示。在这种情况下，ADC 理解为模拟输入电压比实际的值大。

例 11.2　如图 11.25(a)所示是一个 4 位快速 ADC。它由一个和图 11.23 相同的设备进行测试。得到的重构模拟输出如图 11.25(b)所示。判断其中的问题和最有可能出现的错误。

解：缺少的台阶表明 ADC 的输出丢失二进制数 0011。最大的可能是，比较器 3 的输出停留在无效状态(低电平)。

相关问题：假定图 11.25(a)中的 ADC 的比较器 8 停留在高电平输出状态，请在如图 11.23 所示的测试设备中重构模拟输出。

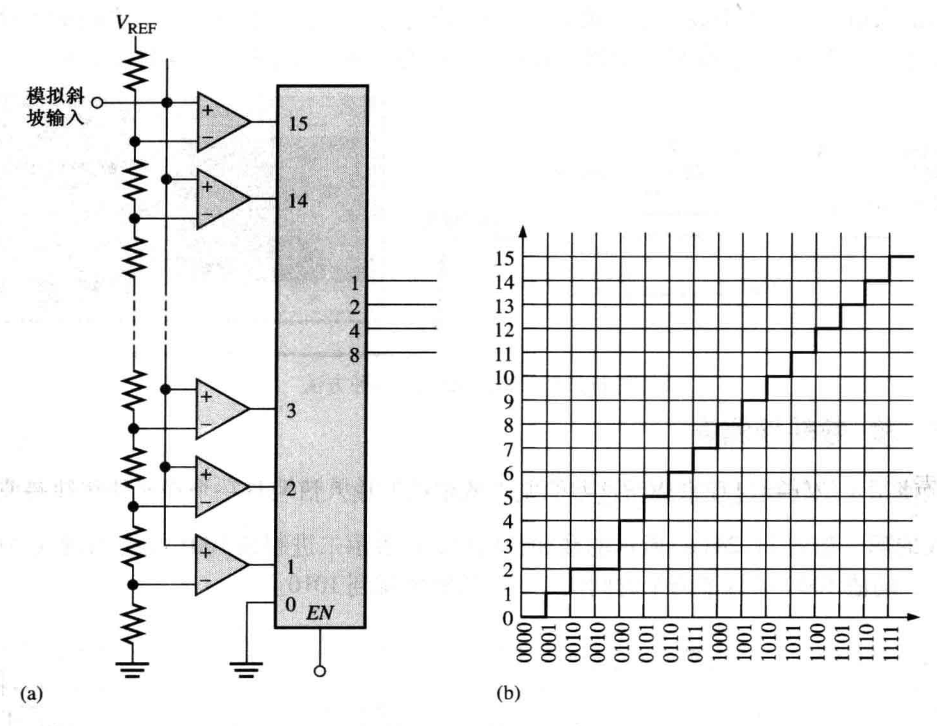


图 11.25

11.2 节 温故而知新

1. 模-数转换的最快的方法是什么?
2. 什么样的模-数转换产生单一位的数据流?
3. 逐次逼近转换器有一个固定的转换时间吗?
4. 给出两种类型的模-数转换器的输出错误。

11.3 数-模转换方法

11.3.1 二进制权输入数-模转换器

数-模转换的一种方法是使用电阻网络，网络中的电阻值表示数字码输入位的二进制权值。图 11.26 给出了一个这种类型的 4 位 DAC。每个输入电阻可以有电流，也可以没有电流，这取决于输入电压的电平。如果输入电压为零(二进制 0)，那么电流也为零。如果输入电压为高电平(二进制 1)，那么电流的大小取决于输入电阻的值，这样对于每个输入电阻其电流不同，如图所示。

因为运算放大器的反向输入(-)几乎没有电流进入，所以所有输入电流将加在一起，并经过电阻 R_f 。因为反向输入为 0 V(虚地)，所以 R_f 两端的电压降等于输出电压，因此 $V_{out} = I_f R_f$ 。

输入电阻值的选择和相应输入位的二进制权值成反比。电阻的最小值(R)对应于最大的二进制权输入(2^3)。其他的电阻值为 R 的倍数(即 $2R$、$4R$ 和 $8R$)，分别对应于二进制权值 2^2、2^1、2^0。输入电流也和二进制权值成正比。因为输入电流的和经过 R_f，所以输出电流与二进制权值的和成正比。

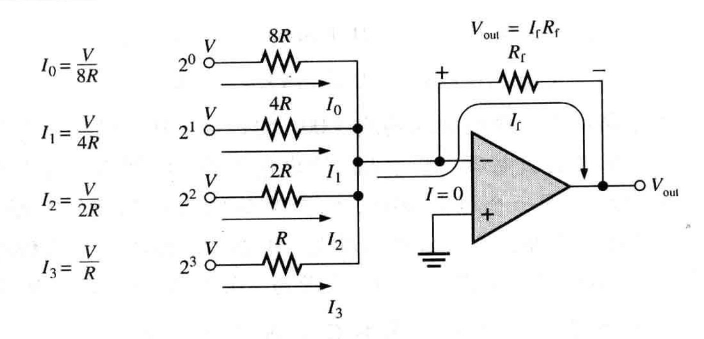


图 11.26　带有二进制权输入的 4 位 DAC

这种类型的 DAC 有缺点，就是不同电阻值的数目较多，所有输入的电压电平必须完全一样。例如，一个 8 位转换器需要 8 个电阻，电阻值的范围以二进制权的步长从 R 到 $128R$。电阻的范围需要 255 分之一（小于 0.5%）的允许误差，才能精确地转换输入，这使得这种类型的 DAC 很难大规模生产。

例 11.3　确定图 11.27(a) 中 DAC 的输出，波形表示在图 11.27(b) 中加到输入的 4 位数的序列。输入 D_0 是最低有效位(LSB)。

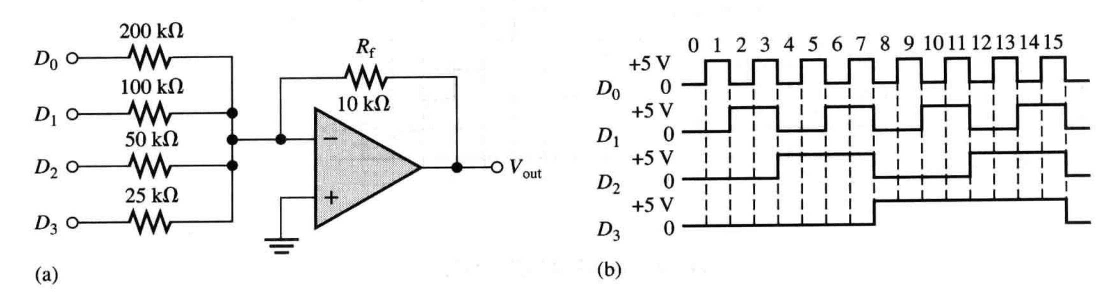


图 11.27

解： 首先，确定每个权输入的电流。因为运算放大器的反向输入（－）是 0 V（虚地），而二进制 1 对应于 +5 V，所以经过任意一个输入电阻的电流是 5 V 除以电阻值。

$$I_0 = \frac{5\ \text{V}}{200\ \text{k}\Omega} = 0.025\ \text{mA}$$

$$I_1 = \frac{5\ \text{V}}{100\ \text{k}\Omega} = 0.05\ \text{mA}$$

$$I_2 = \frac{5\ \text{V}}{50\ \text{k}\Omega} = 0.1\ \text{mA}$$

$$I_3 = \frac{5\ \text{V}}{25\ \text{k}\Omega} = 0.2\ \text{mA}$$

因为反向运算放大器极高的输入阻抗，流进输入的电流几乎没有。因此，假设所有电流都经过反馈电阻 R_f。又因为 R_f 的一端为 0 V（虚地），所以 R_f 两端的电压降等于输出电压，输出电压相对虚拟地来说为负值。

$$V_{out(D0)} = (10\ \text{k}\Omega)(-0.025\ \text{mA}) = -0.25\ \text{V}$$

$$V_{out(D1)} = (10\ \text{k}\Omega)(-0.05\ \text{mA}) = -0.5\ \text{V}$$

$$V_{\text{out}(D2)} = (10\ \text{k}\Omega)(-0.1\ \text{mA}) = -1\ \text{V}$$

$$V_{\text{out}(D3)} = (10\ \text{k}\Omega)(-0.2\ \text{mA}) = -2\ \text{V}$$

从图 11.27(b)可知，第一个二进制输入码是0000，它产生了 0 V 的输出电压。下一个输入码是0001，它产生了 -0.25 V 的输出电压。基于此，输出电压为 -0.25 V。接下来的码是0010，它产生了 -0.5 V 的输出电压。下一个码是 0011，它产生的输出电压是 -0.25 V + (-0.5 V) = -0.75 V。每个连续的二进制数都将使输出电压增大 -0.25 V，因此对于这种特殊的直线形二进制输入序列，输出是从0 ~ -3.75 V 的阶梯形波，步长为 -0.25 V，如图 11.28 所示。

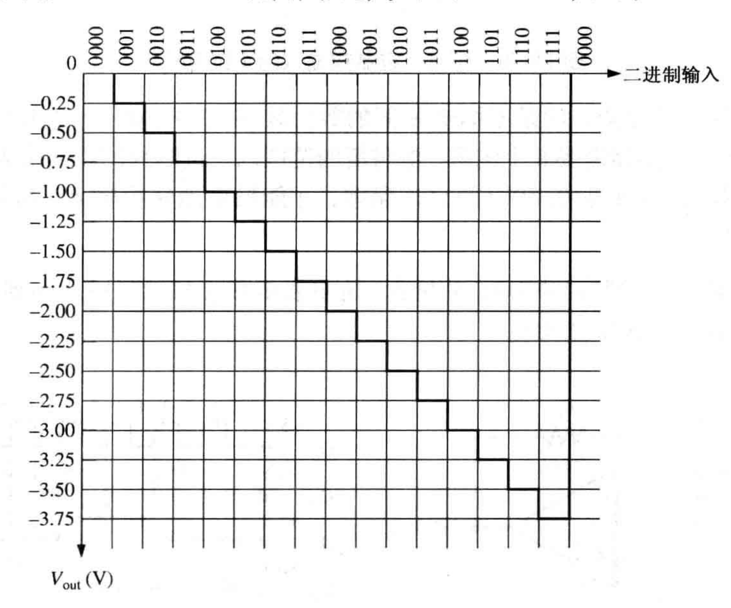


图 11.28　图 11.27 中 DAC 的输出

相关问题：将图 11.27 中到 ADC 的输入波形翻转(D_3到 D_0，D_2到 D_1，D_1到 D_2，D_0到 D_3)，然后确定其输出。

11.3.2　*R*/2*R* 阶梯型数-模转换器

数-模转换的另一种方法是 *R*/2*R* 阶梯型，如图 11.29 所示的四位 DAC。这种方法解决了二进制权输入 DAC 的一个问题，它只需要两个电阻值。

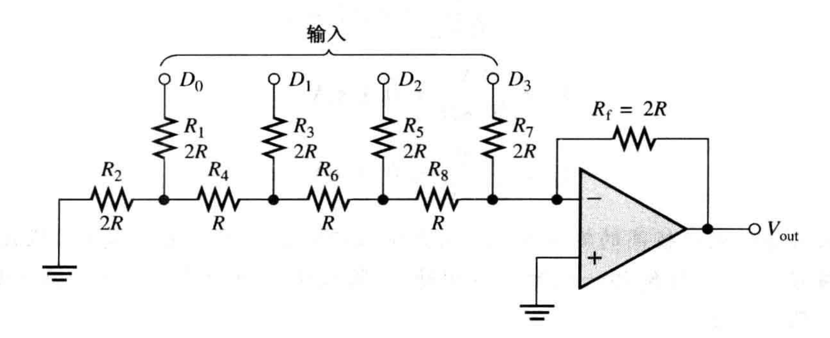
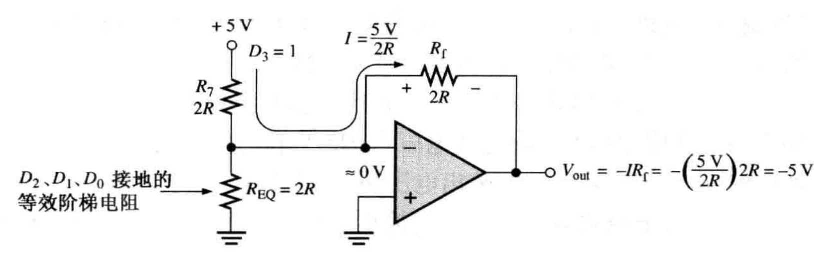
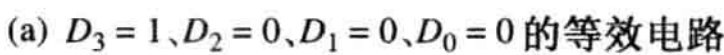
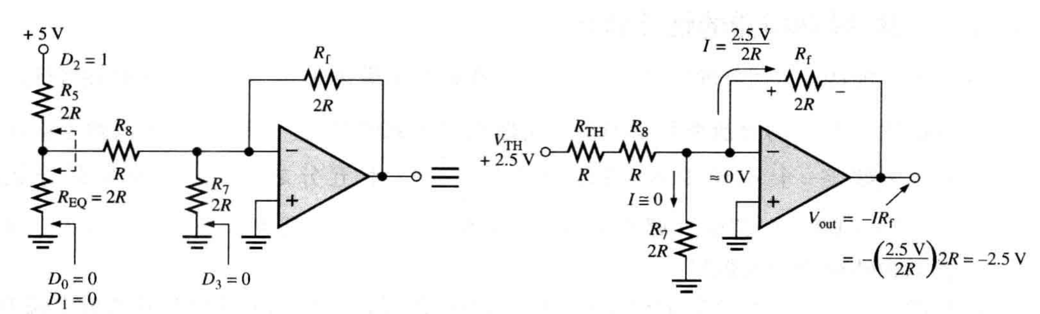


图 11.29　*R*/2*R* 阶梯型 DAC

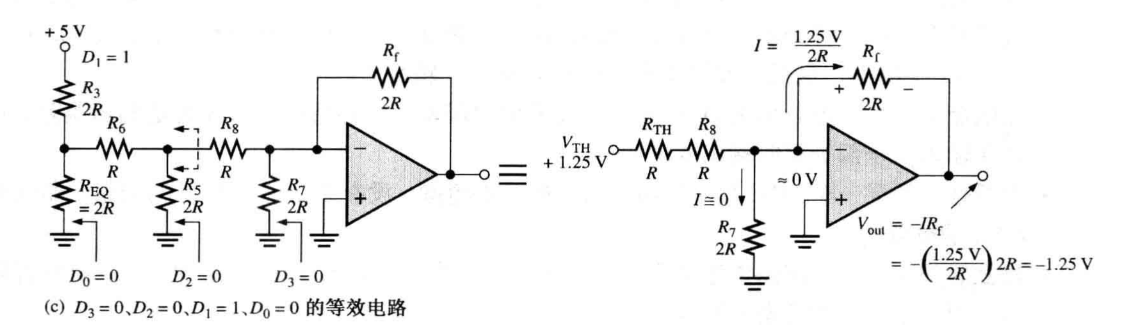
假定开始时输入 D_3 为高电平（$+5\ \mathrm{V}$），其他都为低电平（地，0 V）。这个情况表示二进制数 1000。电路分析指出这种情况能够简化为如图 11.30(a) 所示的等效电路。本质上，没有电流经过 $2R$ 等价电阻（除了 R_7），因为反向输入是虚地。因此，通过 R_7 的电流（$I=5\ \mathrm{V}/2R$）也通过 R_f，而输出电压为 $-5\ \mathrm{V}$。由于负反馈，运算放大器将保持反向输入（−）接近零电压（≈0 V）。因此，所有的电流将通过 R_f，而不是进入反向输入。



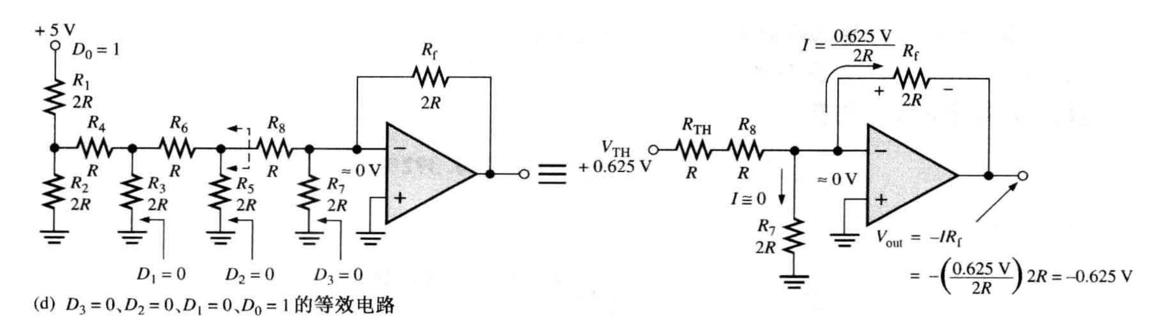

(a) $D_3=1$、$D_2=0$、$D_1=0$、$D_0=0$ 的等效电路



(b) $D_3=0$、$D_2=1$、$D_1=0$、$D_0=0$ 的等效电路



(c) $D_3=0$、$D_2=0$、$D_1=1$、$D_0=0$ 的等效电路



(d) $D_3=0$、$D_2=0$、$D_1=0$、$D_0=1$ 的等效电路

图 11.30　$R/2R$ 阶梯型 DAC 的分析

图 11.30(b)给出了当 D_2 输入等于 +5 V，而其他输入接地时的等效电路。这种情况表示 0100。根据戴维宁定理①，从 R_8 向左看，得到 2.5 V 和与之串联的电阻 R，如图所示。结果得到通过 R_f 的电流 $I = 2.5\ \text{V}/2R$，其输出电压是 −2.5 V。记住，运算放大器的反向输入没有电流进入，也没有电流通过等效的接地电阻，因为虚地的原因，电阻两端的电压为 0 V。

图 11.30(c)给出了当 D_1 输入等于 +5 V，而其他输入接地时的等效电路。这种情况表示 0010。也是根据戴维宁定理，从 R_8 向左看，得到 1.25 V 和与之串联的电阻 R，如图所示。结果得到通过 R_f 的电流 $I = 1.25\ \text{V}/2R$，其输出电压是 −1.25 V。

图 11.30(d)给出了当 D_0 输入等于 +5 V 而其他输入接地时的等效电路。这种情况表示 0001。从 R_8 向左看的戴维宁电路给出等效电压 0.625 V 和与之串联的电阻 R，如图所示。结果得到通过 R_f 的电流 $I = 0.625\ \text{V}/2R$，其输出电压是 −0.625 V。

注意，每个连续的较低权输入产生减半的输出电压，这样，输出电压就与输入位的二进制权值成正比。

11.3.3　数-模转换器的性能特点

DAC 的性能特点包括分辨率、精度、线性度、单调性及建立时间，下面对每种特性进行讨论：

- **分辨率**。DAC 的分辨率就是输出中离散的增量数的倒数。当然，这是依赖于输入位的数目。例如，一个 4 位 DAC 的分辨率是 $\frac{1}{2^4-1}$（十五分之一）。表示为百分数，就是 $(1/15)\cdot 100 = 6.67\%$。离散阶梯数的总数等于 2^n-1，其中 n 是位的数目。分辨率还可表示为转换的位的数目。
- **精度**。精度是 DAC 实际输出和期望输出相比较而得到的，表示为满量程或最大输出电压的百分数。例如，一个转换器的满量程输出为 10 V，精度为 ±0.1%，那么任何输出电压的最大误差为(10 V)(0.001) = 10 mV。理想情况下，精度应当不低于最低有效位的 ±1/2。对于一个 8 位转换器，最低有效位是满量程的 0.39%。精度大约为 ±0.2%。
- **线性度**。线性误差是偏离 DAC 理想直线输出的误差。一种特殊的情况就是偏移误差，它是在输入位全部为零时输出的电压值。
- **单调性**。如果一个 DAC 顺序经历输入位的全部范围，没有获得任何相反的增量，那么此 DAC 是单调的。
- **建立时间**。建立时间通常定义为从输入码发生变化，到 DAC 在最终值的 ±1/2 最低有效位以内稳定下来所花费的时间。

例 11.4　确定下列情况的分辨率，表示为百分数：

(a)8 位 DAC　(b)12 位 DAC

解：(a)对于 8 位转换器，

$$\frac{1}{2^8-1}\times 100 = \frac{1}{255}\times 100 = \mathbf{0.392\%}$$

(b)对于 12 位转换器，

$$\frac{1}{2^{12}-1}\times 100 = \frac{1}{4095}\times 100 = \mathbf{0.0244\%}$$

相关问题：计算 16 位 DAC 的分辨率。

---

① 戴维宁定理：任何电路可以等效为一个电压源串联一个等效电阻。

11.3.4　测试数-模转换器

DAC 测试的概念如图 11.31 所示。在这个基本方法中，在输入加一个二进制码序列，观察得到的输出。二进制码序列扩展到满量程，即从 $0 \sim 2^n - 1$ 升序排列的全部值，其中 n 是位的数目。

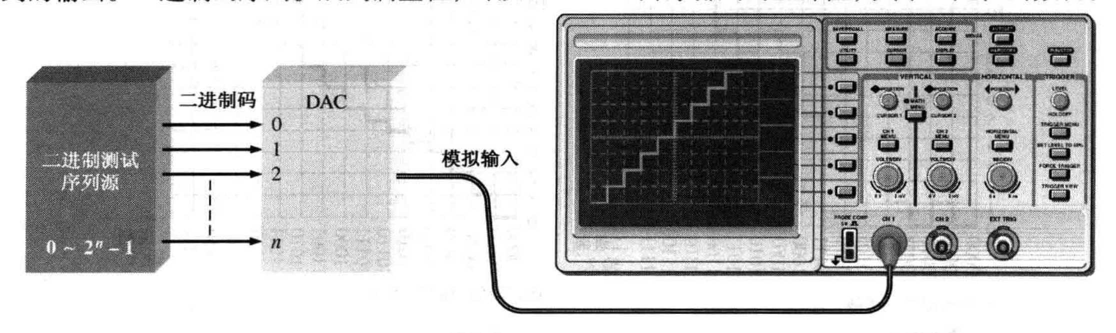


图 11.31　DAC 的基本测试设备

理想的输出是如图所示的直线形阶梯。随着二进制码的位数增加，分辨率也将提高。也就是说，随着离散增量的数量增加，输出更接近于直线线性斜坡。

11.3.5　数-模转换错误

图 11.32 给出了几种检测到的数-模转换错误，为了举例说明使用 4 位转换。4 位转换将产生 15 步离散的增量。图中的每个图形都包含一条理想的阶梯状斜坡，作为错误输出的比较。

非单调性　图 11.32(a)中颠倒的增量表示了非单调性，这是非线性的一种。在这种特殊的情况下，出现错误是因为二进制编码中的 2^1 位被解释为常数 0。也就是说，一个短路引起输入线路停留在低电平。

非线性偏差　图 11.32(b)给出了非线性偏差的情况，对于某些输入码，增量的幅度小于它应该有的大小。这种特殊的输出可能是由于 2^2 位的权值不够大，也可能是由于不正确的输入电阻。如果一个特殊的二进制权值大于它应该具有的值，那么增量的幅度也大于它应该具有的值。

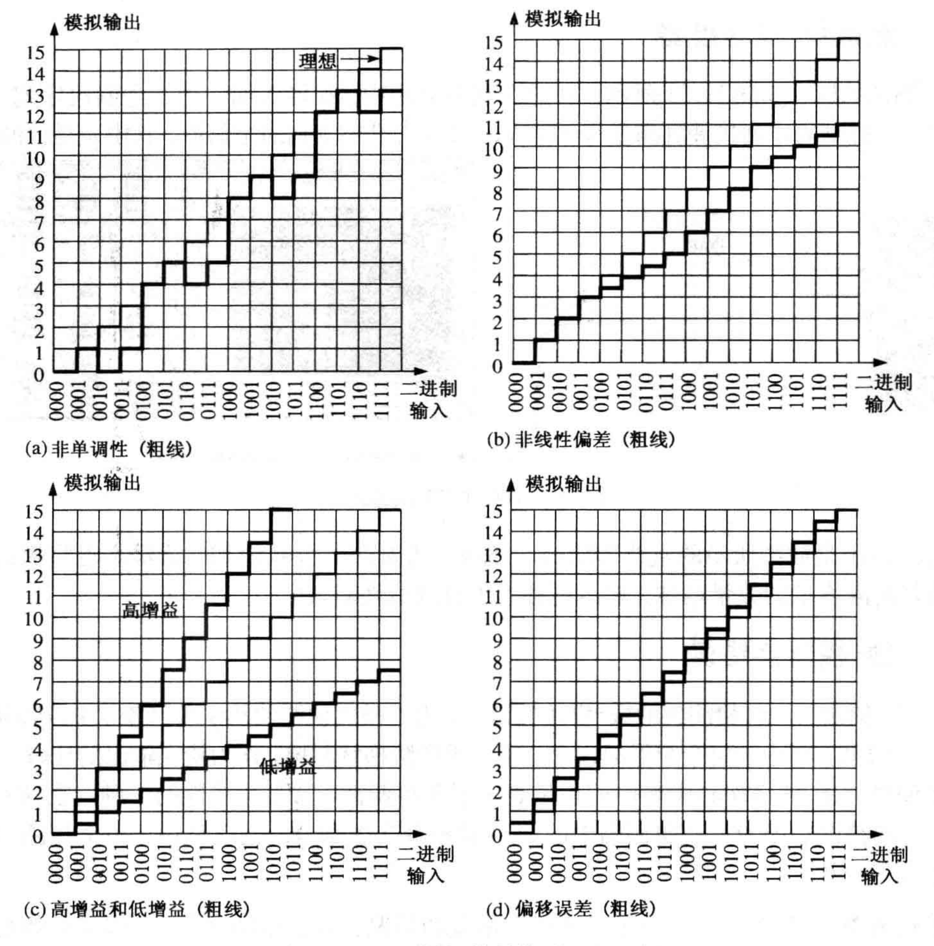
低或高增益　由低或高增益引起的输出错误如图 11.32(c)所示。在低增益的情况下，所有增量的幅度都小于理想值。在高增益的情况下，所有增量的幅度都大于理想值。这种情形可能是由运算放大器电路中不正确的反馈电阻引起的。

偏移误差　偏移误差如图 11.32(d)所示。注意，当二进制输入为 0000 时，输出电压不是零，而且这个偏移量的值对于转换中的所有增量都是相同的。不正确的运算放大器可能是产生这种情况的根本原因。

例 11.5　图 11.33 为一个直线形 4 位二进制序列加到输入后所观察到的 DAC 的输出。判断错误类型，并给出避免错误的方法。

解：这种情况下的 DAC 是非单调的。分析输出发现设备转换成为下面的序列，而不是实际施加在输入上的二进制序列。

0010，0011，0010，0011，0110，0111，0110，0111，1010，1011，1010，1011，1110，1111，1110，1111



(a) 非单调性（粗线）

(b) 非线性偏差（粗线）

(c) 高增益和低增益（粗线）

(d) 偏移误差（粗线）

图 11.32　几种数-模转换错误的图例

显然，2^1位停留在高电平(1)状态。要找到问题，首先应监视设备输入引脚的状态。如果正在改变状态，那么是 DAC 内部错误，应当将它替换掉。如果外部引脚没有改变状态，并且总是为高电平，那么可能是由于电路板上某个焊接物的跨接造成 + V 和外部短路。

相关问题：若直线形的 4 位二进制序列加在输入，并且 2^0 位保持为高电平，确定 DAC 的输出。

11.3.6　重构滤波器

DAC 的输出是一个“阶梯”状的波形，近似于经过数字信号处理器处理后的原始模拟信号。低通重构滤波器(有时称为后滤波器)的用途是消除较高的高频成分，以平滑 DAC 的输出，高频成分是“阶梯”的快速转换导致的，图 11.34 进行了大致的描述。

11.3 节　温故而知新

1. 具有二进制权输入的 DAC(数-模转换器)的优点是什么？
2. 4 位 DAC 的分辨率是多少？
3. 如何检测 DAC 中的非单调性？
4. DAC 输出低增益的效果是什么？

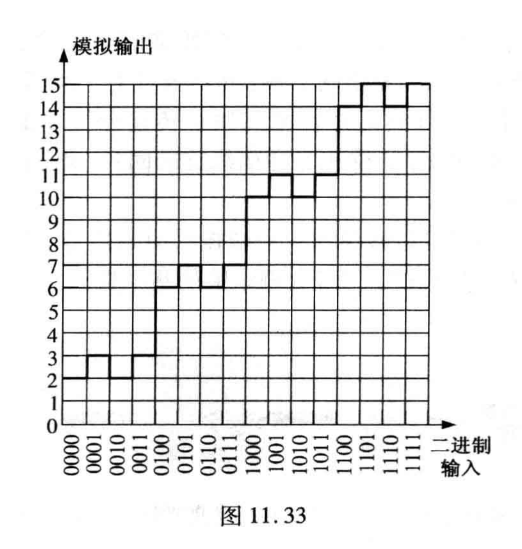


图 11.33

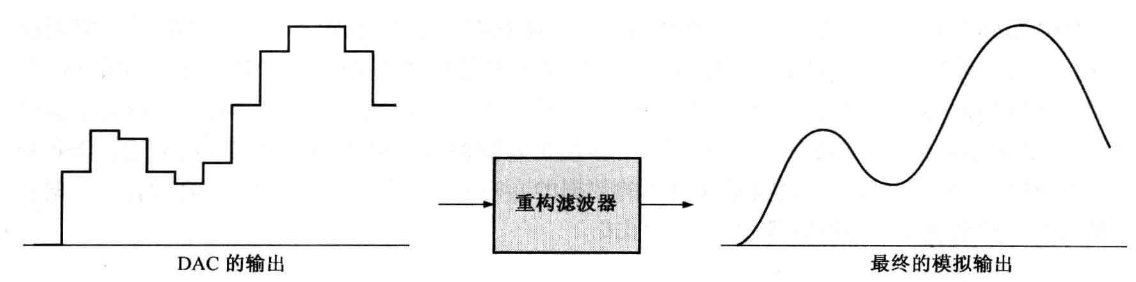


图 11.34 重构滤波器平滑 DAC 的输出

11.4 数字信号处理基础

数字信号处理系统实现把连续变化的模拟信号转换为一系列离散的电平。这些离散的电平大小的变化和模拟信号的变化同步，如图 11.35 所示的正弦波的情况。把原来的模拟信号变为阶梯信号的过程是由采样和保持电路完成的。

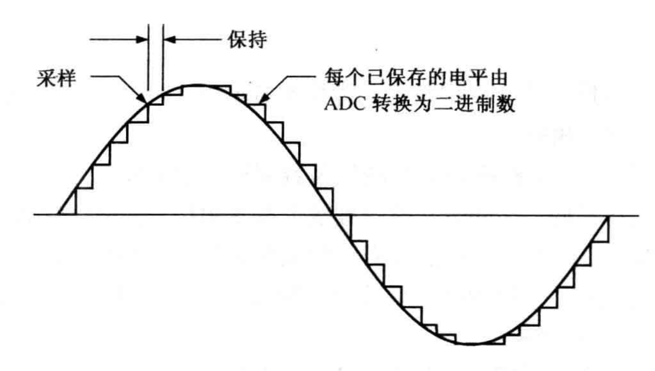


图 11.35 一个原始的模拟信号(正弦波)和它的"阶梯"近似电平

接下来，"阶梯"近似电平被量化为表示离散阶梯电平的二进制数，这样的过程称为模 - 数转换。完成此模-数转换的电路就是模-数转换器(ADC)。

一旦模拟信号被转换为二进制的形式，就被加到数字信号处理器(DSP)中。DSP可以对输入的数据进行各种运算，例如去除不需要的干扰，增大某个信号频率的幅度并减少其他信号频率的幅度，为安全传输的目的进行编码并检测和纠正传输得到的数码。在其他许多运算中，DSP还可以对录制的声音进行净化，去除来自通信线路的回音，增强用于医疗诊断的CT扫描的图像和用于手机加扰的转换。

在DSP处理信号之后，信号可以转换回一个增强的原始信号。这由数-模转换器(DAC)来完成。图11.36给出了一个典型的数字信号处理系统的基本框图。

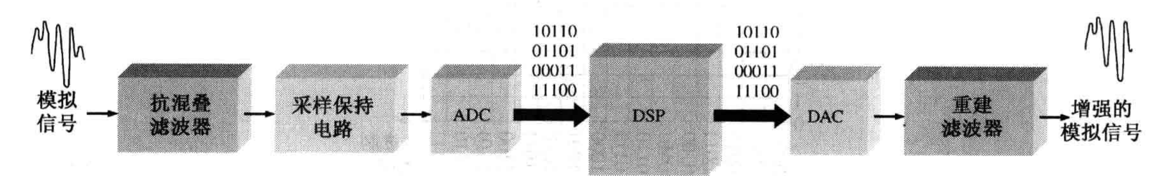


图11.36 一个典型的信号处理系统的基本框图

DSP实际上就是一个特殊类型的微处理器，但是在两个重要方面它和微处理器的一般用途不同。通常，设计微处理器用于一般功能，并使用大型软件进行操作。DSP用于具体的应用，它是一个高速的数字计算器，使用具体的算法(程序)实时处理获取的信息。在一个系统中的模-数转换器必须对输入的模拟信号进行采样，推出足以捕捉信号幅度的所有相关的波动，DSP必须跟上ADC的采样速度，在接收采样得到的数据的同时进行运算。一旦DSP处理数据，数据就会进入数-模转换器，转换回模拟信号的形式。

11.4节 温故而知新

1. DSP表示什么?
2. ADC表示什么?
3. DAC表示什么?
4. 使用哪种电路把一个模拟信号转换为二进制数?
5. 使用哪种电路把一个二进制数转换为模拟形式?

关键词

混叠 当一个信号以低于信号频率两倍的速度采样时所产生的效应。混叠产生不需要的信号频率，干扰接收到的信号频率。

模-数转换器(ADC) 用于把模拟信号转换为数字形式的电路。

解码 DSP流水线运行的一个步骤，指令被赋予基本单位，并进行解码。

数-模转换器(DAC) 用于把模拟信号表示的数字转换回模拟信号的电路。

DSP 数字信号处理器，一种特殊类型的微处理器，实时处理数据。

DSP核 DSP的中央处理单元。

执行指令 DSP流水线运行的一个步骤，进行指令的译码。

取指令 DSP流水线运行的一个步骤，从程序存储器中获取指令。

MFLOPS 每秒一百万次的浮点运算。

MIPS 每秒一百万次的指令。

MMACS　每秒一百万次乘/累加。

奈奎斯特频率　在以具体采样频率采样时，可以采样的最高频率；此最高频率小于采样频率的一半。

流水线　DSP 体系结构的一部分，允许多条指令同时处理。

量化　在模-数转换中，每个采样值被赋予一个二进制数的过程。

采样　获取一个波的足够多的离散值，从而定义这个波的形状的过程。

判断题 （答案在本章的结尾。）

1. 由 ADC 把一个模拟信号转换为一个数字信号。
2. 一个 ADC 近似于一个计算机。
3. 奈奎斯特频率是采样频率的两倍。
4. 对于一个给出的模拟信号，高采样速率比低采样速率精确。
5. 分辨率是一个模 - 数转换器使用的位的数目。
6. 逐次逼近是一种模-数转换的方法。
7. 增量调制基于两个连续采样的差。
8. DAC 的两个类型是二进制权输入和 *R*/2*R* 梯形。
9. 一个模拟值转换为码的过程称为量化。
10. 一个快速 ADC 来自于一个同步 ADC。

自测题 （答案在本章的结尾。）

1. ADC 是
   (a)字母数据编码器　(b)模-数转换器　(c)模拟设备载波器　(d)模数比较器
2. DAC 是
   (a)数模计算机　(b)数字分析计算器　(c)数据累积转换器　(d)数-模转换器
3. 模拟信号的采样产生
   (a)一系列与信号幅度成正比的脉冲　(b)一系列与信号频率成正比的脉冲
   (c)表示模拟信号幅度的数字码　(d)表示每次采样时间的数字码
4. 根据采样定理，采样频率应当
   (a)小于最高信号频率的一半　(b)大于最高信号频率的两倍
   (c)小于最低信号频率的一半　(d)大于最低信号频率
5. 保持操作出现在
   (a)每次采样前　(b)每次采样期间　(c)模-数转换后　(d)紧跟在采样后
6. 量化过程是
   (a)把采样和保持输出转换成二进制
   (b)把采样脉冲转换成一个电平
   (c)把二进制码序列转换成重构的模拟信号
   (d)在采样发生前滤除不需要的频率
7. 一般情况下，模拟信号可通过下列哪种方式更准确地重构?
   (a)更多的量化电平　(b)更少的量化电平　(c)更高的采样频率　(d)更低的采样频率
   (e)(a)或(c)都可以
8. 快速 ADC 使用

(a)计数器 (b)运算放大器 (c)积分器 (d)触发器
(e)(a)和(c)

9. 双积分 ADC 使用
(a)计数器 (b)运算放大器 (c)积分器 (d)微分器
(e)(a)和(c)

10. 求和-增量 ADC 的输出是
(a)并行二进制码 (b)多位数据 (c)单一的位数据 (d)差分电压

11. 在二进制权 DAC 中，输入的电阻将
(a)决定模拟信号的振幅 (b)决定数字输入的权值
(c)限制功率损耗 (d)阻止从数据源载入

12. 在一个 $R/2R$ DAC 中，有
(a)4 个电阻值 (b)1 个电阻值
(c)2 个电阻值 (d)电阻值的数量等于输入的数量

13. 一个数字信号系统通常运行在
(a)实时 (b)虚拟时间 (c)压缩时间 (d)计算机时间

习题 (奇数题的答案在本书的结尾。)

11.1 节 模拟信号转换为数字信号

1. 图 11.37 所示的波形加在采样电路上，每 3 ms 采样一次。给出采样电路的输出。假设在输入和输出之间是一对一的电压对应关系。

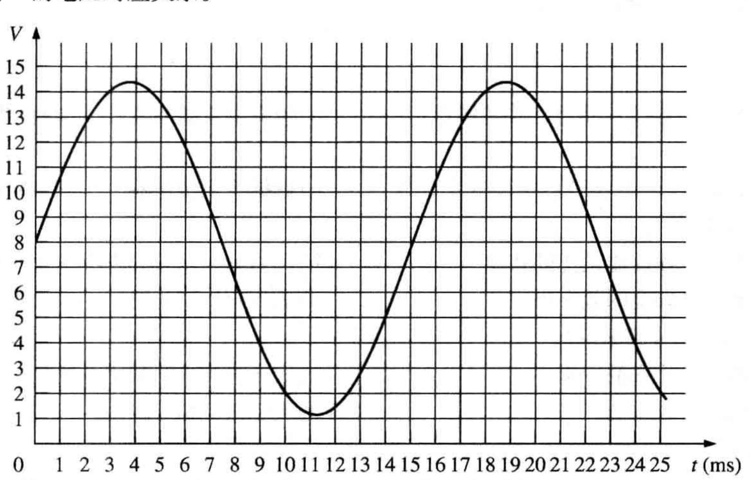


图 11.37

2. 把习题 1 中采样电路的输出加在保持电路上。给出保持电路的输出。
3. 如果将习题 2 中保持电路的输出进行两位量化，那么得到的二进制码序列是怎样的？
4. 请使用 4 位量化重复习题 3。
5. (a)以习题 3 中的 2 位量化序列重构模拟信号。
(b)以习题 4 中的 4 位量化序列重构模拟信号。
6. 画出由下列二进制数序列表示的模拟函数。
1111, 1110, 1101, 1100, 1010, 1001, 1000, 0111, 0110, 0101, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1100, 1100, 1011, 1010, 1001.

11.2 节　模-数转换方法

7. 某个运算放大器组成的反向放大器的输入电压是 10 mV，输出是 2 V。其闭环电压增益是多少？
8. 要使某个反向放大器的闭环电压增益达到 330，如果 R_i = 1.0 kΩ，那么反馈电阻应该是多少？
9. 如果输入电阻为 2.2 kΩ，那么使用 47 kΩ 反馈电阻的反相放大器的增益是多少？
10. 对于一个 8 位快速模-数转换器，需要多少个比较器？
11. 对于如图 11.38 所示的模拟输入信号，确定一个 3 位快速 ADC 的二进制输出码。

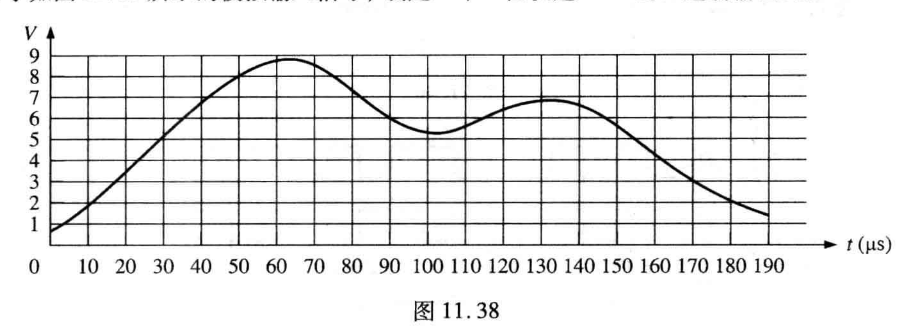


图 11.38

12. 针对图 11.39 所示的模拟波形重复习题 11。

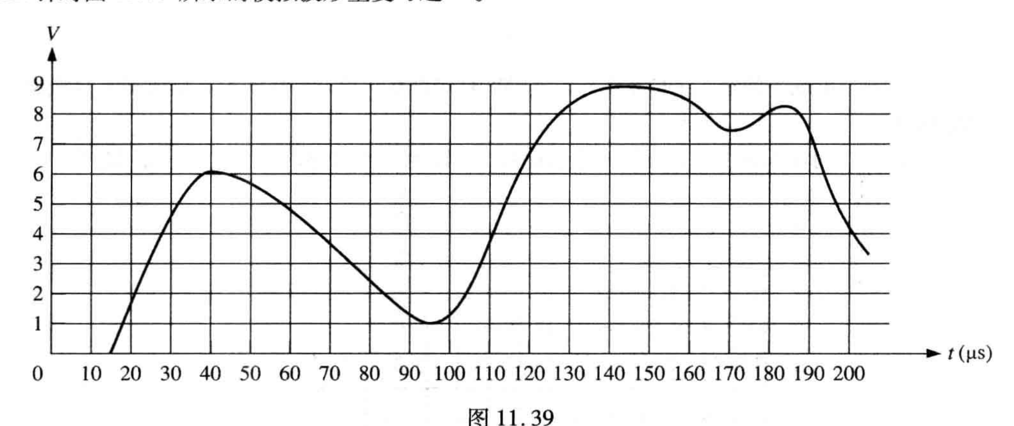


图 11.39

13. 对某个 2 位逐次渐近 ADC，其最大的阶梯输出是 +8 V。如果给模拟输入加上恒定的 +6 V 电压，确定逐次渐近寄存器(SAR)的二进制状态序列。
14. 针对 4 位逐次渐近 ADC，重复习题 13。
15. 当一个模拟信号加在 ADC 的输入上时，它产生了下列二进制数序列：

    0000，0001，0010，0011，0100，0101，0110，0111，0110，0101，0100，0011，0010，0001，0000.

    (a) 以数字方式重构输入；

    (b) 如果 ADC 工作出现问题，丢失了码 0111，那么重构的输出是怎样的？

11.3 节　数-模转换方法

16. 在图 11.26 的 4 位 DAC 中，最低权电阻的值是 10 kΩ。其他输入电阻应该是多少？
17. 如果给输入加上如图 11.40(b) 所示的 4 位数字序列，确定图 11.40(a) 的 DAC 的输出。数据输入的低电压为 0 V，高电压为 +5 V。
18. 针对图 11.41 中的输入，重复习题 17。
19. 请针对下列每一种 DAC，确定其以百分数表示的分辨率：

    (a) 3 位　　(b) 10 位　　(c) 18 位
20. 为图 11.31 所示的测试设备开发一个可产生 8 位二进制测试序列的电路。

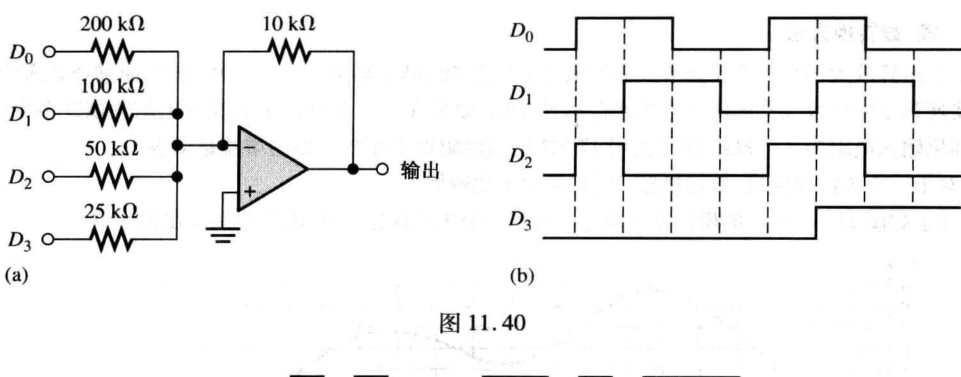


图 11.40

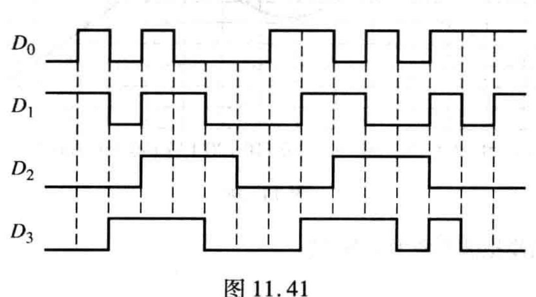


图 11.41

21. 一个4位DAC出现了问题，它的最高有效位(MSB)停留在0状态。绘制给其输入加上直线形二进制序列时的模拟输出。

22. 一个4位DAC的输入加上了直线形二进制序列，其输出如图11.42所示。有什么问题?

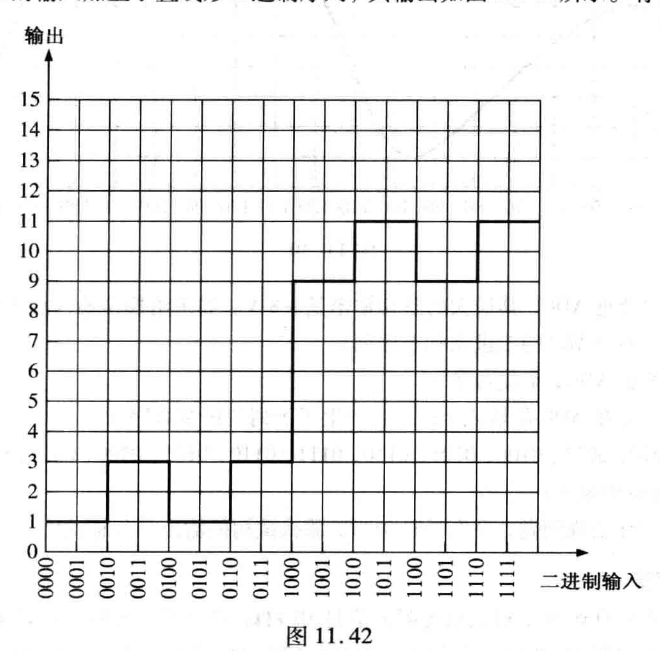


图 11.42

11.4 节 数字信号处理基础

23. 解释模-数转换的目的。

24. 为图11.43的数字信号处理框图填上合适的功能名称。

25. 解释数-模转换的目的。

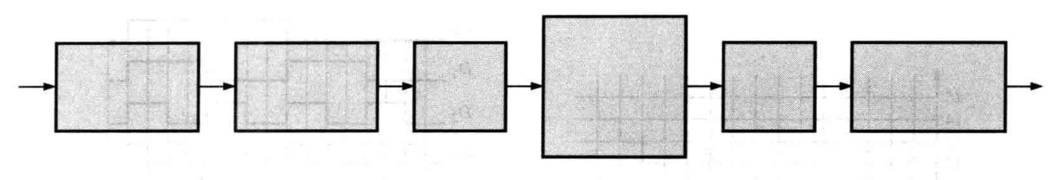


图 11.43

答案

温故而知新

11.1 节　模拟信号转换为数字信号

1. 采样是把模拟信号转换成一系列的脉冲的过程，每个脉冲对应于模拟信号的幅度。
2. 保持采样值是为了有时间把它转换为二进制码。
3. 最小的采样频率是 40 kHz。
4. 量化是一个把采样电平转换为二进制码的过程。
5. 位的数目确定了量化的精度。

11.2 节　模-数转换方法

1. 同步(快速)方法最快。
2. 求和-增量方法产生一个单一位的数据流。
3. 是的，逐次逼近有固定的转换时间。
4. 丢失码、不正确的码和偏移是 ADC 输出错误的类型。

11.3 节　数-模转换方法

1. 在二进制权 DAC 中，每个电阻的阻值不同。
2. $(1/(2^4-1)))\times 100\% = 6.67\%$。
3. 在 DAC 中的一个相反的步骤表示非单调行为。
4. 在 DAC 中幅度的间距在低增益时比理想的要小。

11.4 节　数字信号处理基础

1. DSP 表示数字信号处理器。
2. ADC 表示模-数转换器。
3. DAC 表示数-模转换器。
4. ADC 把模拟信号转换为二进制信号的形式。
5. DAC 把二进制信号转换为模拟信号的形式。

例题的相关问题

11.1　100，111，100，000，011，110。是，信息丢失。

11.2　参见图 11.44。

11.3　参见图 11.45。

11.4　$\left(\frac{1}{2^{16}-1}\right)\times 100\% = 0.00153\%$。

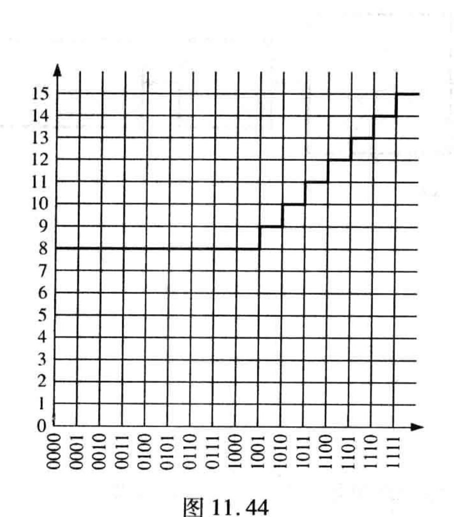


图 11.44

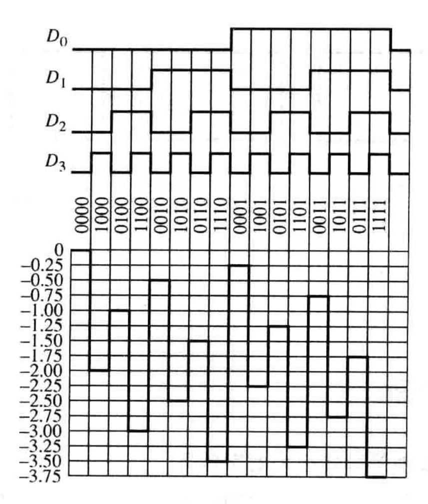


图 11.45

判断题

1. T　2. F　3. F　4. T　5. T　6. F　7. T　8. T　9. T　10. F

自测题

1. (b)　2. (d)　3. (a)　4. (b)　5. (d)　6. (a)　7. (e)
8. (b)　9. (e)　10. (c)　11. (b)　12. (c)　13. (a)

第12章　集成电路技术

章节提纲

12.1　基本操作特性和参数

12.1.1　DC供电电压

TTL(晶体管-晶体管逻辑)芯片名义上的直流供电电压是+5 V。TTL也称为T^2L。CMOS(互补金属氧化物半导体)芯片在+5 V和+3.3 V、2.5 V和1.2 V的不同电压下均可工作。为了简化起见，直流供电电压在逻辑框图中被省略了，但它其实是连接在IC封装芯片的V_{CC}引脚上的，而地则连接GND引脚。电压和地在内部分配给IC封装芯片中的所有元件，图12.1是一个14引脚封装芯片。

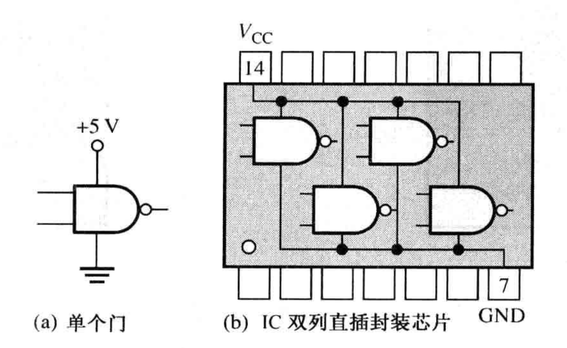


图12.1　V_{CC}和地的连接及在IC封装芯片中的分配情况。为了简化，忽略了其他引脚的连接

12.1.2　CMOS逻辑电平

第1章中简要介绍了逻辑电平。有4种不同的逻辑电平规范：V_{IL}，V_{IH}，V_{OL}，V_{OH}。对于CMOS电路来说，输入电压的范围(V_{IL})可以用一个有效低电平(逻辑0)表示，这个有效低电平对于5 V逻辑是从0 V到1.5 V，对于3.3 V逻辑是从0 V到0.8 V。表示有效高电平(逻辑1)的输入电压的范围(V_{IH})，对于5 V逻辑是3.5 V到5 V，对于3.3 V逻辑是2 V到3.3 V，如图12.2所示。对于5 V逻辑，1.5 V到3.5 V，以及对于3.3 V逻辑，0.8 V到2 V的电压值范围是不可预测性能的区域，在这些区域的电压值是不允许出现的。当输入电压在这些区域时，逻辑电路也许将其解释为高电平或低电平。因此，当输入电压在这些不允许出现的区域时，CMOS门电路的操作是不可靠的。

CMOS 输出电压的范围(V_{OL}和 V_{OH})对应于5 V 和3.3 V 逻辑电平如图 12.2 所示。注意，输出电压的最小高电平 $V_{OH(min)}$ 比输入电压的最小高电平 $V_{IH(min)}$ 要大。同样还要注意，输出电压的最大低电平值 $V_{OL(max)}$ 比输入电压的最大低电平 $V_{IL(max)}$ 要小。

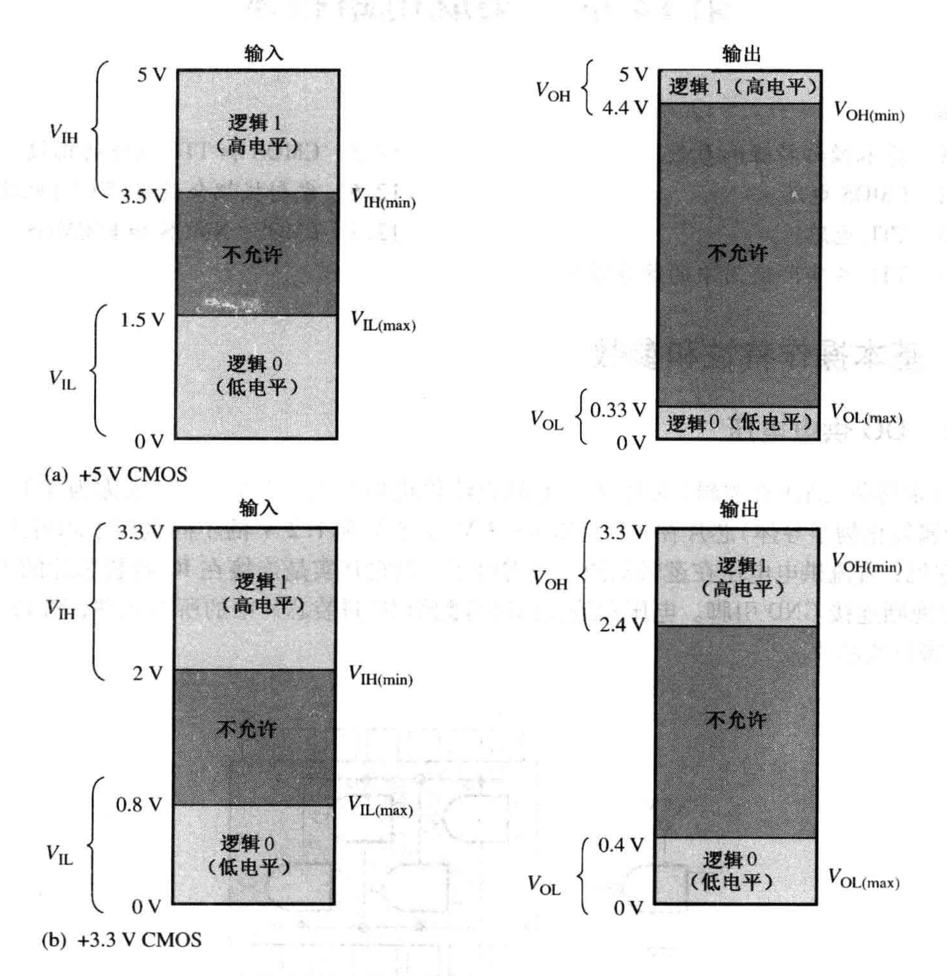


图 12.2 CMOS 的输入和输出逻辑电平

12.1.3 TTL 逻辑电平

TTL 电路的输入和输出逻辑电平如图 12.3 所示。与 CMOS 类似，它也有 4 种不同的逻辑电平规范：V_{IL}，V_{IH}，V_{OL}，V_{OH}。

12.1.4 抗噪度

噪声是电路中产生的一种不需要的电压，噪声可能威胁到电路的正常工作。系统中的导线和其他导体可能受到杂散的高频电磁辐射的干扰，这些电磁辐射来自邻近的导体，这些导体中的电流正在快速变化，或者来自许多系统之外的其他干扰源。此外，输电线的电压波动是一种低频噪声。

为了避免噪声的不利影响，逻辑电路必须具有一定的抗噪度。这是一种能力，即在输入端容许有一定程度的不需要的电压波动，并且没有改变它的输出状态。例如，在高电平状态下，

如果噪声电压导致 5 V CMOS 门电路的输入降到了 3.5 V 之下，那么输入电压就会在不允许的范围内，这时操作是不可预测的(见图 12.2)。这样，门电路也许把低于 3.5 V 的波动解释为低电平，如图 12.4(a)所示。同样，如果噪声导致门电路的输入在低电平状态下高于 1.5 V，那么就会出现不确定的情况，如图 12.4(b)所示。

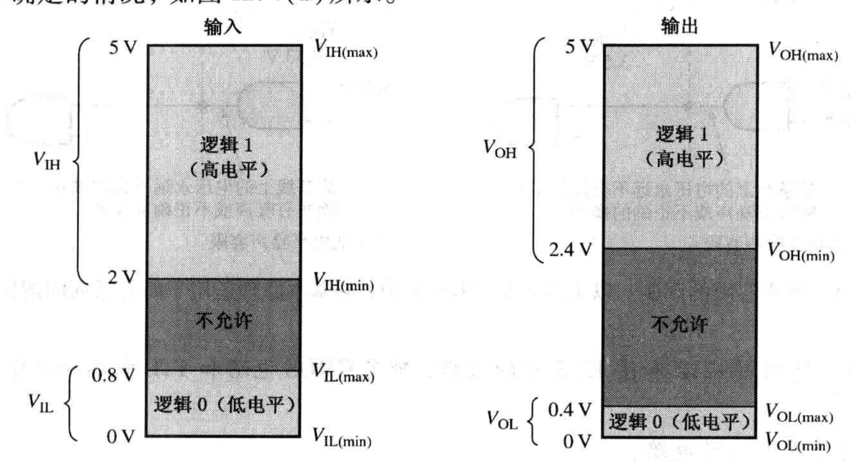


图 12.3　TTL 输入和输出的逻辑电平

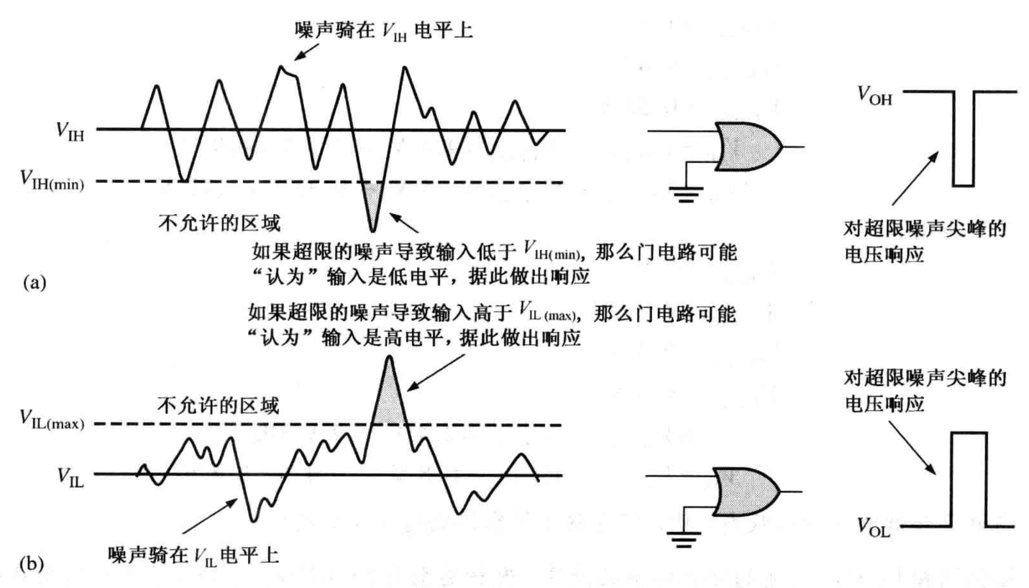


图 12.4　门电路工作中输入噪声的影响的示意图

12.1.5　噪声容限

电路抗噪度的一个衡量标准叫做**噪声容限**(noise margin)，用伏特表示。对于一个给定的逻辑电路，有两个噪声容限值：噪声容限高电平(V_{NH})和噪声容限低电平(V_{NL})。这两个参数由如下的公式定义：

$$V_{NH} = V_{OH(min)} - V_{IH(min)} \quad (12.1)$$

$$V_{NL} = V_{IL(max)} - V_{OL(max)} \quad (12.2)$$

有时看到噪声容限是用 V_{CC} 的百分比来表示的。从上面公式可以看出，V_{NH} 是电压差，是驱动门高电平输出的最小值($V_{OH(min)}$)和负载门能够允许的高电平输入的最小值($V_{IH(min)}$)之差。噪声容

限 V_{NL}是门电路能够容许的低电平输入的最大值($V_{IL(max)}$)和驱动门电路可能的低电平输出的最大值($V_{OL(max)}$)两者之差。噪声容限如图 12.5 所示。

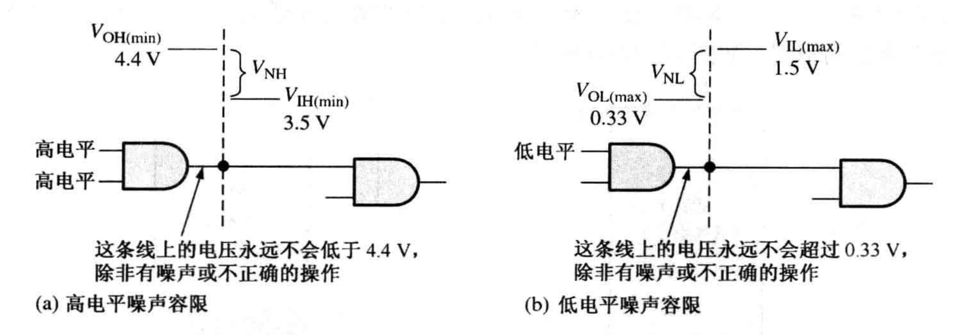


图 12.5　噪声容限的图例。以上是 5 V CMOS 的值，但基本原理适用于所有系列的逻辑电路

例 12.1　使用图 12.2 和图 12.3 中的数据，确定 CMOS 电路和 TTL 电路的高电平与低电平噪声容限。

解：对于 5 V CMOS 电路，

$$V_{IH(min)} = 3.5\ V$$
$$V_{IL(max)} = 1.5\ V$$
$$V_{OH(min)} = 4.4\ V$$
$$V_{OL(max)} = 0.33\ V$$
$$V_{NH} = V_{OH(min)} - V_{IH(min)} = 4.4\ V - 3.5\ V = \mathbf{0.9\ V}$$
$$V_{NL} = V_{IL(max)} - V_{OL(max)} = 1.5\ V - 0.33\ V = \mathbf{1.17\ V}$$

对于 TTL 电路，

$$V_{IH(min)} = 2\ V$$
$$V_{IL(max)} = 0.8\ V$$
$$V_{OH(min)} = 2.4\ V$$
$$V_{OL(max)} = 0.4\ V$$
$$V_{NH} = V_{OH(min)} - V_{IH(min)} = 2.4\ V - 2\ V = \mathbf{0.4\ V}$$
$$V_{NL} = V_{IL(max)} - V_{OL(max)} = 0.8\ V - 0.4\ V = \mathbf{0.4\ V}$$

对于高电平和低电平输入状态，TTL 门电路能够容许高达 0.4 V 的噪声。

相关问题[①]：根据上述的噪声容限的计算，哪种类型的 5V CMOS 或 TTL 芯片应该用在高噪声的环境中？

12.1.6　功耗

逻辑门从直流电源获得电流，如图 12.6 所示。当门电路处于高电平输出状态时，一个称为 I_{CCH}的总电流流出；而当门电路处于低电平输出状态时，另一个总电流 I_{CCL}流出。

作为一个例子，当 V_{CC}为 5 V 时，I_{CCH}设定为 1.5 mA，并且如果门电路处于静态(不变化)的高电平输出状态，那么该门电路的功耗(P_D)为

① 答案在本章的结尾。

$$P_D = V_{CC}I_{CCH} = (5\ \text{V}) \times (1.5\ \text{mA}) = 7.5\ \text{mW}$$

当门电路受脉冲作用时，其输出将在高电平和低电平之间交替切换，所产生的供电电流将在 I_{CCH}和 I_{CCL}之间变化。平均功耗取决于占空比，占空比一般指定为50%。当占空比为50%时，那么输出在一半的时间内为高电平，而在另一半的时间内为低电平。因此平均的供电电流为

$$I_{CC}\ \frac{I_{CCH} + I_{CCL}}{2} \tag{12.3}$$

平均功耗为

$$P_D = V_{CC}I_{CC} \tag{12.4}$$

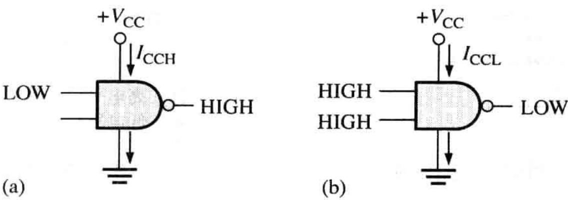


图 12.6　来自直流电源的电流。如图为传统的电流方向，电子流向的标记相反

例 12.2　某个门电路，当输出为高电平时，流入 2 μA 的电流；当输出为低电平时，流入 3.6 μA的电流。如果 V_{CC}是 5 V，该门电路的占空比是 50%，那么平均功耗是多少？

解： 平均电流 I_{CC}为

$$I_{CC}\ \frac{I_{CCH} + I_{CCL}}{2} = \frac{2.0\ \mu\text{A} + 3.6\ \mu\text{A}}{2} = 2.8\ \mu\text{A}$$

平均功耗为

$$P_D = V_{CC}I_{CC} = (5\ \text{V})(2.8\ \mu\text{A}) = \mathbf{14\ \mu W}$$

相关问题： 一个 IC 门电路的 $I_{CCH} = 1.5\ \mu\text{A}$ 和 $I_{CCL} = 2.8\ \mu\text{A}$。确定在 V_{CC}是 5 V、占空比是 50%的情况下，门电路的平均功耗是多少？

TTL 电路的功耗在工作频率的范围以内基本上是恒定的。但是，CMOS 电路的功耗则和频率有关。在静态（直流）条件下它的功耗是非常小的，并随着频率的增加而增加。这些特性如图 12.7 中总的曲线所示。例如，一个低功耗肖特基（LS）TTL 门电路的功耗是常量 2.2 mW。一个 CMOS 门电路的功耗在静态条件下是 2.75 μW，在频率为 100 kHz 时是 170 μW。

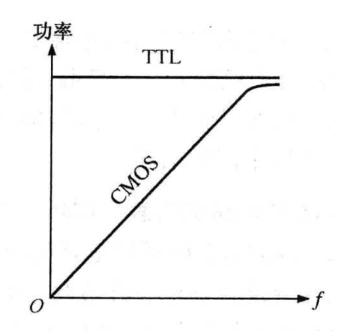


图 12.7　TTL 和 CMOS 功率-频率曲线

12.1.7　传输延迟时间

当一个信号在逻辑电路中通过（传送）时，总会经历一个时间延迟，如图 12.8 所示。输入电平的变化引起输出电平的变化，输出电平变化的发生总是迟后输入一个很短的时间，这称为**传输延迟时间**。

正如第 3 章所提及的，逻辑门规定了两个传输延迟时间：

- t_{PHL}：当输出从高电平向低电平变化时，输入脉冲的指定点与输出脉冲的对应点之间的时间。

- t_{PLH}：当输出从低电平向高电平变化时，输入脉冲的指定点与输出脉冲的对应点之间的时间。

这两种传输延迟时间如图 12.9 所示，以脉冲边沿的 50% 的点作为参照点。

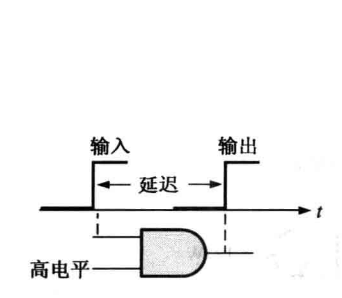


图 12.8 传输延迟时间的基本图例

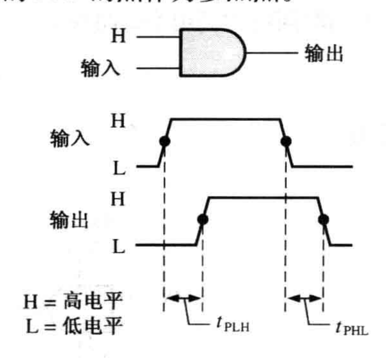


图 12.9 传输延迟时间

门电路的传输延迟时间限制了它所能够工作的频率。传输延迟时间越长，最高频率就越低。因此，速度较高的电路的传输延迟时间就较短。例如，传输延迟为 3 ns 的门电路就比传输延迟为 10 ns 的门电路的速度快。

12.1.8 速度-功率乘积

在为某种应用选择所使用的逻辑电路类型时，如果传输延迟时间和功耗都是重要的考虑因素，那么速度-功率乘积提供了各种逻辑电路的比较基础。速度-功率乘积越低越好。速度-功率乘积的单位是微微焦耳(pJ)。例如，HCMOS 电路在 100 kHz 的频率下，其速度-功率乘积为 1.2 pJ，而对应的 LS TTL 的速度-功率乘积则为 22 pJ。

12.1.9 负载和扇出

当一个逻辑门的输出连接到了一个或多个其他门的输入时，驱动门上就会有一个负载，如图 12.10 所示。对于一个给定的门，它可以驱动的负载门输入是有数量限制的。这个限制就称为这个门的扇出。

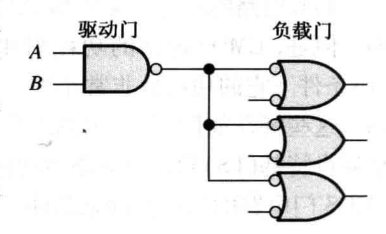


图 12.10 带门输入负载的门输出

CMOS 电路的负载 CMOS 电路的负载不同于 TTL 电路的负载，因为 CMOS 逻辑电路所使用的晶体管为驱动门提供了一种占主导地位的电容性负载，如图 12.11 所示。在这种情况下，限制条件是与驱动门输出电阻和负载门输入电容相关的充电和放电时间。当驱动门的输出是高电平时，负载门的输入电容通过驱动门的输出电阻充电。当驱动门电路的输出是低电平时，电容放电，如图 12.11 所示。

如果给驱动门的输出添加更多的负载门输入，那么总的电容将会增加，因为输入电容实际上是并行出现的。电容的增加会延长充电和放电时间，因此就会降低这个门电路的最大工作频率。因此，CMOS 门电路的扇出是和工作频率有关的。负载门输入越少，最大频率就越高。

TTL 负载 TTL 驱动门向处于高电平状态的负载门提供灌电流(I_{IH})，从处于低电平状态的

负载门吸收拉电流(I_{IL})。灌电流和拉电流如图 12.12 所示，其中的电阻代表两种情况下门内部的输入和输出电阻。

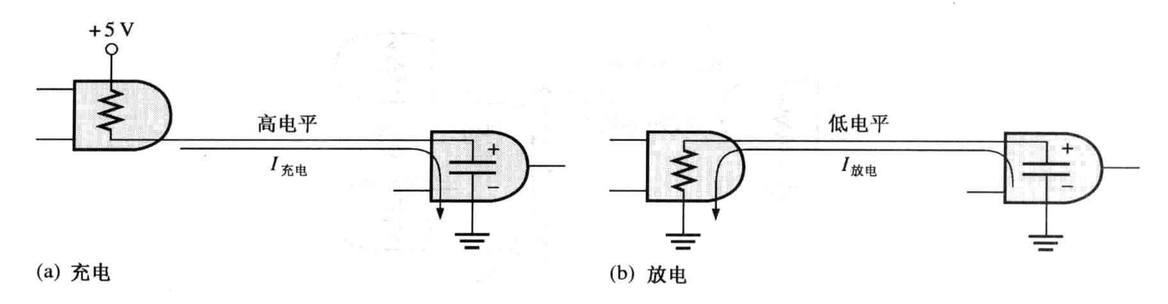


图 12.11　CMOS 门的容性负载

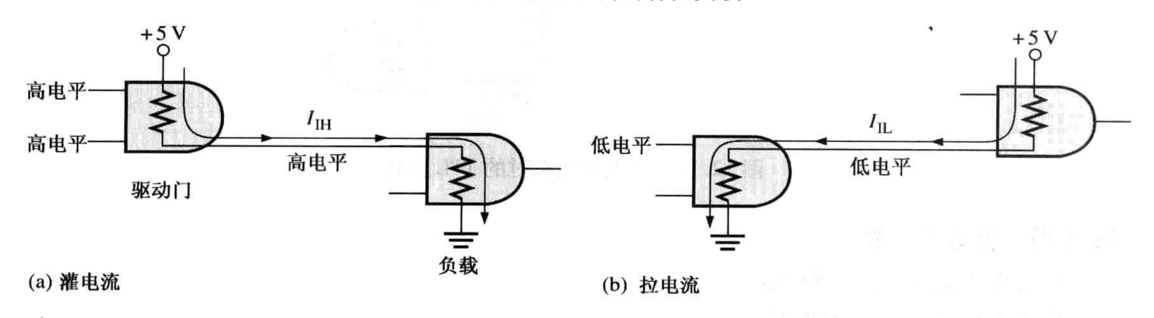


图 12.12　逻辑门中灌电流和拉电流的基本图例

当更多的负载门连接到驱动门时，驱动门电路上的负载就会增加。总的灌电流随着每增加一个门输入而增加，如图 12.13 所示。随着电流的增加，驱动门内部的电压降也会增加，从而引起输出电压 V_{OH} 的降低。如果连接过多的负载门输入，V_{OH} 就会降到低于 $V_{OH(min)}$，高电平噪声容限也会降低，从而影响电路的正常工作。同样，随着灌电流的增加，驱动门的功耗也会增加。

扇出是指在不会对门的工作特性造成不利影响的情况下，门所能够连接的负载门输入的最大数目。例如，低功耗肖特基(LS)TTL 电路的扇出是 20 个单位负载。作为驱动电路的相同逻辑系列的一个输入称为一个单位负载。

总的拉电流也会随着每增加一个负载门输入而增加，如图 12.14 所示。当拉电流增加时，驱动门电路内部的电压降也会增加，从而导致 V_{OL} 增加。如果添加了过多数的负载，那么 V_{OL} 将会超过 $V_{OL(max)}$，而低电平噪声容限会减小。

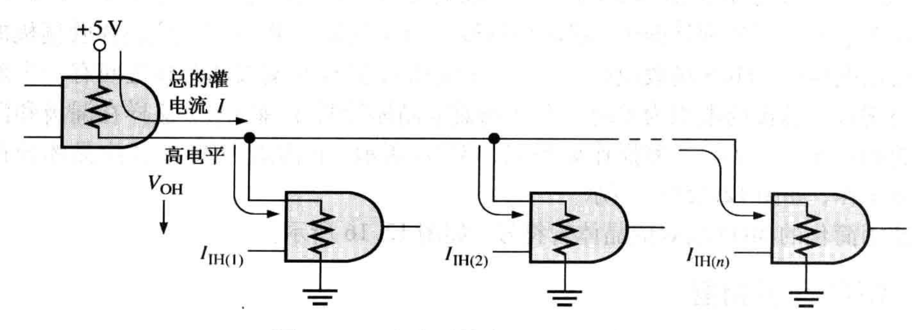


图 12.13　高电平状态下的 TTL 负载

在 TTL 电路中，拉电流的能力(低电平输出状态)是决定扇出数量的限制因素。

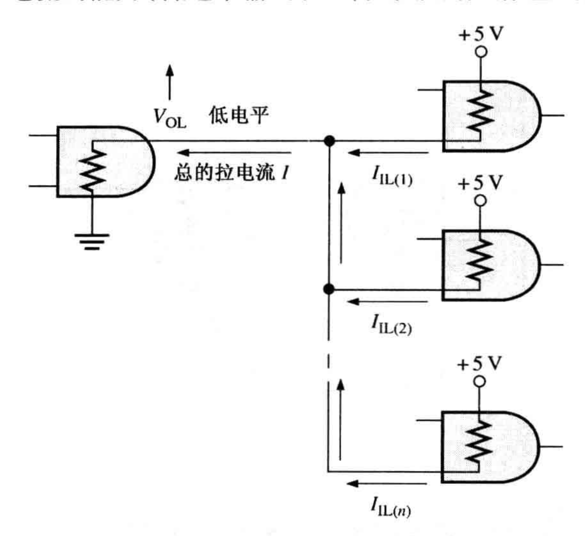


图 12.14 低电平状态时的 TTL 负载

12.1 节 温故而知新

1. 定义 V_{IH}、V_{IL}、V_{OH}和 V_{OL}。
2. 噪声容限的值低好还是高好?
3. 门 A 比门 B 的传输延迟时间长，哪一个门可以工作在更高的频率下?
4. 过多的负载是如何影响噪声容限的?

12.2 CMOS 电路

12.2.1 MOSFET

金属氧化物半导体场效应晶体管(MOSFET)是 CMOS 电路中的有源开关元件。这些芯片在结构和内部操作方面与 TTL 电路中使用的双极型晶体管有很大的差别，但开关的作用基本上是一样的：根据输入电平，它们将作为理想的打开或闭合开关。

图 12.15(a)给出了 n 沟道和 p 沟道 MOS 场效应晶体管的符号。如图所示，一个 MOS 场效应晶体管的 3 个引出端分别是栅极、漏极和源极。当 n 型沟道 MOS 场效应晶体管栅极的正电压高于源极的正电压时，MOS 场效应晶体管导通(饱和)，这样在漏极和源极之间有一个理想的闭合开关。当栅极到源极的电压为零时，MOS 场效应晶体管断开(截止)，这样在漏极和源极之间有一个理想的打开的开关。这种操作如图 12.15(b)所示。p 沟道 MOS 场效应晶体管在相反的电压极性下工作，如图 12.15(c)所示。

有时使用简化的 MOS 场效应晶体管符号，如图 12.16 所示。

12.2.2 CMOS 反相器

互补 MOS(CMOS)逻辑把互补对中的 MOS 场效应晶体管用做基本单元。一个互补对使用 p 沟道和 n 沟道增强型 MOS 场效应晶体管，如图 12.17 的反相器电路所示。

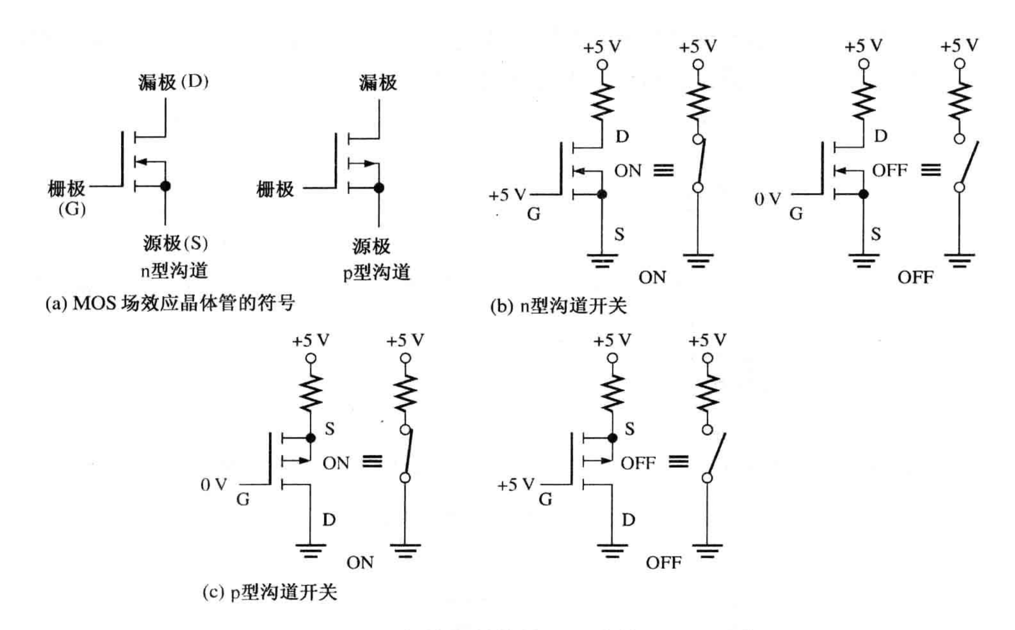


图 12.15　MOS 场效应晶体管的基本符号和开关作用

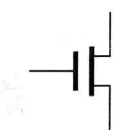


图 12.16　简化的 MOS 场效应晶体管符号

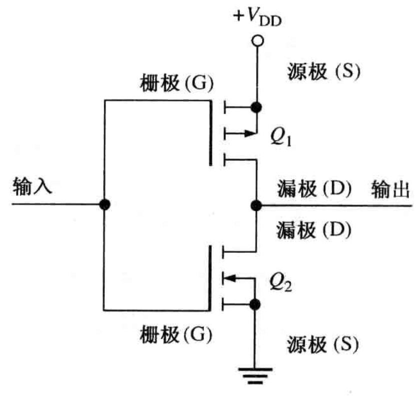


图 12.17　一个 CMOS 反相器电路

当一个高电平加在输入时，如图 12.18(a)所示，p 沟道 MOS 场效应晶体管 Q_1截止，n 沟道 MOS 场效应晶体管 Q_2导通。这个情况通过 Q_2的导通电阻使输出连接到地，导致低电平输出。当一个低电平加在输入时，如图 12.18(b)所示，Q_1导通，Q_2截止。这个情况通过 Q_1的导通电阻把输出连接到 $+V_{DD}$，导致高电平输出。

12.2.3　CMOS 与非门电路

图 12.19 给出了一个有两个输入的 CMOS 与非门电路。注意互补对(n 沟道和 p 沟道 MOS 场效应晶体管)的排列。

CMOS 与非门电路的工作如下所述。

- 当两个输入都是低电平时，Q_1和 Q_2导通，Q_3和 Q_4截止。通过 Q_1和 Q_2并联的导通电阻，输出被上拉为高电平。
- 当输入 A 为低电平、输入 B 为高电平时，Q_1和 Q_4导通，Q_2和 Q_3截止。通过 Q_1的低导通电阻，输出被上拉为高电平。
- 当输入 A 为高电平、输入 B 为低电平时，Q_1和 Q_4截止，Q_2和 Q_3导通。通过 Q_2的低导通电阻，输出被上拉为高电平。

- 最后，当两个输入都为高电平时，Q_1和Q_2截止，Q_3和Q_4导通。在这个情况下，通过Q_3和Q_4的串联导通电阻的接地，输出被下拉到低电平。

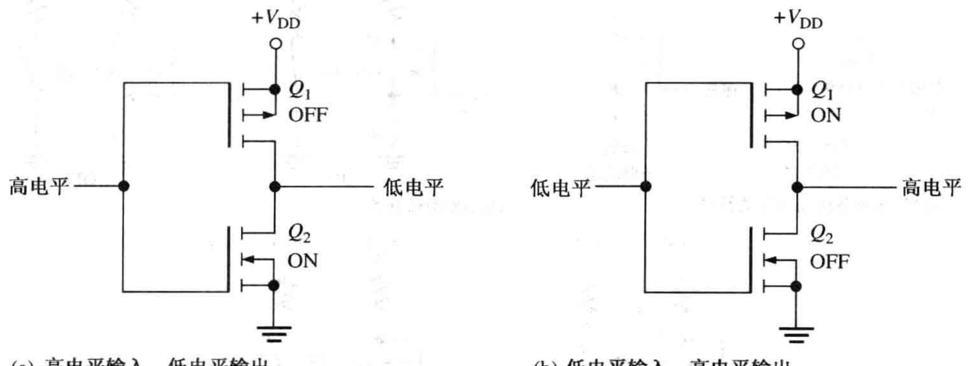


图 12.18 CMOS 反相器的工作原理

+V_DD

Q_1 Q_2 Q_3 Q_4

输出

输入 A

输入 B

A	B	Q_1	Q_2	Q_3	Q_4	X
L	L	S	S	C	C	H
L	H	S	C	C	S	H
H	L	C	S	S	C	H
H	H	C	C	S	S	L

C＝截止（断开） H＝高电平

S＝饱和（导通） L＝低电平

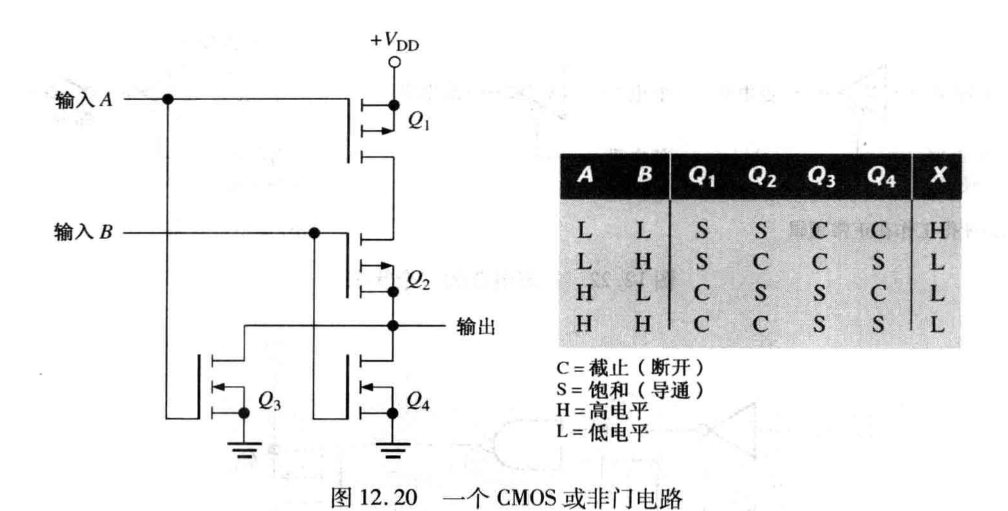
图 12.19 一个 CMOS 与非门电路

12.2.4 CMOS 或非门

图 12.20 给出了一个有两个输入的 CMOS 或非门。注意互补对的排列。

CMOS 或非门电路的操作如下所述。

- 当两个输入都是低电平时，Q_1和Q_2导通，Q_3和Q_4截止。结果，通过Q_1和Q_2导通电阻的串联，输出被上拉为高电平。
- 当输入A为低电平、输入B为高电平时，Q_1和Q_4导通，Q_2和Q_3截止。通过Q_4的低导通电阻接地，输出被下拉到低电平。
- 当输入A为高电平、输入B为低电平时，Q_1和Q_4截止，Q_2和Q_3导通。通过Q_3的导通电阻接地，输出被下拉到低电平。
- 当两个输入都是低电平时，Q_1和Q_2截止，Q_3和Q_4导通。通过Q_3和Q_4并联的导通电阻接地，输出被下拉到低电平。



A	B	Q_1	Q_2	Q_3	Q_4	X
L	L	S	S	C	C	H
L	H	S	C	C	S	L
H	L	C	S	S	C	L
H	H	C	C	S	S	L

图 12.20　一个 CMOS 或非门电路

12.2.5　漏极开路门

术语“漏极开路”的意思是输出晶体管的漏极端没有连接，必须从外部通过负载连接到 V_{DD}。漏极开路门是集电极开路 TTL 门(下一节讨论)的 CMOS 芯片。漏极开路输出电路是一个单一的 n 沟道 MOS 场效应晶体管，如图 12.21(a)所示。为了产生高电平输出，必须使用一个外部上拉电阻，如图 12.21(b)所示。此外，漏极开路输出能够以线与的方法连接，这个概念将在下一节涉及 TTL 时讨论。

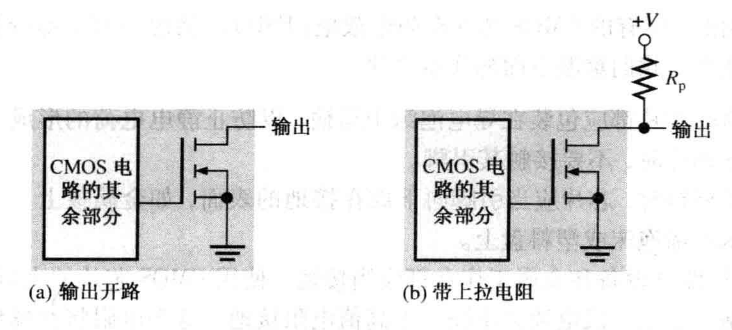


图 12.21　漏极开路 CMOS 门

12.2.6　CMOS 三态门

三态输出在 CMOS 和 TTL 逻辑电路中都是有效的。三态输出结合了推拉输出和集电极开路电路的优点。回顾一下，三种输出状态是高电平、低电平和高阻抗(高 Z)。由使能 ($\overline{Enable}$) 输入的状态决定三态门的输出状态，当选择工作在正常的逻辑电平时，三态电路和常规门的工作方式相同。当三态电路选择工作在高阻状态时，通过内部电路把三态电路的输出和其余部分有效地断开。图 12.22 给出了三态电路的操作。反三角(∇)表示三态输出。

在 CMOS 三态门的电路系统中，如图 12.23 所示，允许把每个输出晶体管 Q_1 和 Q_2 同时关闭，从而断开了输出和电路其余部分的连接。

当使能输入为低电平时，使得电路工作在正常的逻辑状态。当使能输入为高电平时，Q_1 和 Q_2 都截止，电路处于高阻抗状态。

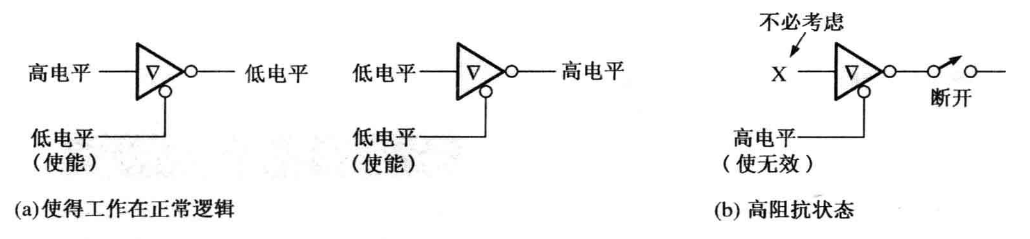


图 12.22　三态电路的三个状态

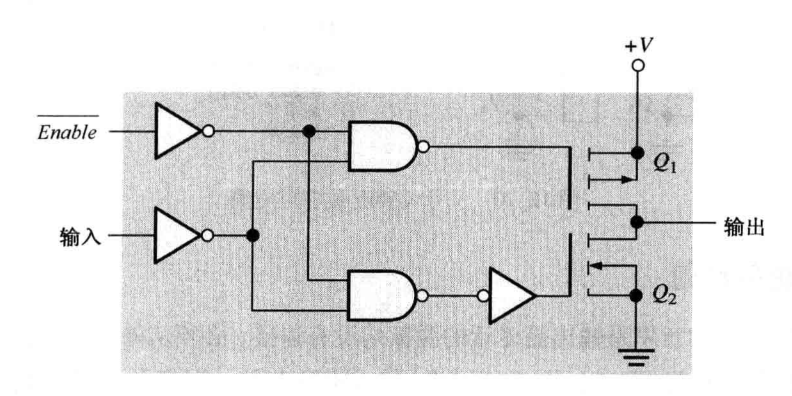


图 12.23　三态 CMOS 反相器

12.2.7　使用 CMOS 芯片的注意事项

正如已经学到的，所有的 CMOS 芯片在静电放电(ESD)的情况下都会遭受损害，因此在使用它们时要特别注意。我们重温下面的注意事项：

1. 所有的 CMOS 芯片都应包装在导电泡沫中运输，以防止静电电荷的形成。从泡沫包装中拿出 CMOS 芯片时，不要接触其引脚。
2. 在撤走保护材料时，芯片应当引脚向下放在接地的表面，如金属板上。不要将 CMOS 设备放在聚苯乙烯泡沫或塑料盘上。
3. 所有的工具、测试设备和金属工作台都应当接地。使用 CMOS 芯片的人员在某些环境中，应当在手腕上加上一段电缆并串联一个高值电阻接地。这个电阻将在操作人员接触电压源时防止强烈的电击。
4. 不要把 CMOS 芯片(或任何其他 IC)插入通电的插槽或 PCB 上。
5. 所有不使用的输入都应该连接到电源或地，如图 12.24 所示。如果不接，输入就可能获得静电电荷，从而在不可预测的电平上“漂浮”。
6. 在 PCB 上装配好后，在存储或运输时，把芯片的连接口插在导电泡沫上，以提供必要的保护。用高阻值的电阻把 CMOS 的引脚与地连接，这样引脚也可以得到保护。

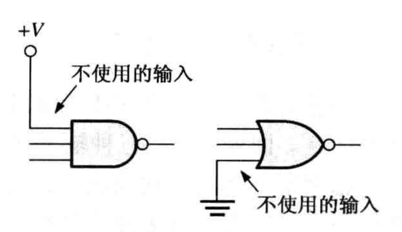


图 12.24　CMOS 不使用的输入的处理

12.2 节　温故而知新

1. 在 CMOS 中使用什么类型的晶体管？
2. 术语互补 MOS 是什么意思？
3. 为什么 CMOS 芯片必须小心处理？

12.3　TTL 电路

12.3.1　双极型晶体管(BJT)

双极型晶体管(BJT)是所有 TTL 电路中使用的有源开关元件。图 12.25 所示为一个 npn 双极型晶体管的符号，它有 3 个引出端：基极、发射极和集电极。双极型晶体管有两个结，即基极-发射极结和基极-集电极结。

基本的开关工作原理如下：若基极的正电压高于发射极电压 0.7 V 左右，有足够的电流提供给基极，这时晶体管导通，进入饱和状态。在饱和状态下，在集电极和发射极之间，晶体管的作用就像一个理想的闭合的开关，如图 12.26(a)所示。当基极的正电压大于发射极电压的值小于 0.7 V 时，晶体管截止，在集电极和发射极之间变成一个打开的开关，如图 12.26(b)所示。用通常的术语总结，就是基极上的高电平使晶体管导通，使得它成为一个闭合开关；基极上的低电平使晶体管截止，使得它成为一个打开的开关。一般意义的总结，就是基极上的一个高电平使晶体管导通，即把开关合上；基极上的一个低电平使晶体管截止，即把开关打开。在 TTL 电路中，一些双极型晶体管有多个发射极。

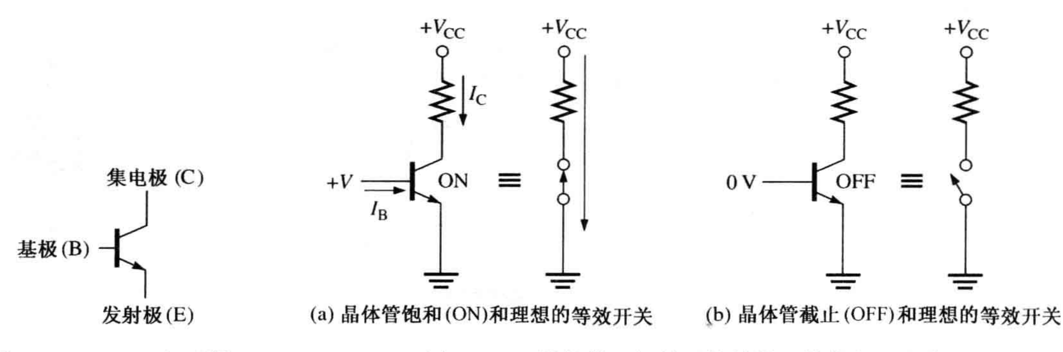


图 12.25　npn 双极型晶体管的符号

图 12.26　晶体管理想的开关特性。传统的电流方向如图所示。电子流动方向的标记相反

12.3.2　TTL 反相器

无论使用哪种电路技术，反相器或者任何类型的门，其逻辑功能总是相同的。图 12.27 给出了反相器的标准 TTL 电路。在图中 Q_1 是输入耦合晶体管，D_1 是输入钳位二极管。晶体管 Q_2 称为相位“分离器”，Q_3 和 Q_4 的组合形成输出电路，通常称为推拉输出(图腾柱，totem-pole)。

当输入为高电平时，Q_1 的基极-发射极结是反向偏置，而基极-集电极结是正向偏置。这个情况允许电流通过 R_1 和 Q_1 的基极-集电极结进入 Q_2 的基极，从而驱动 Q_2 进入饱和状态。结果，Q_2 的导通导致 Q_3 导通，Q_3 的集电极电压就是输出，它接近于接地电压。这样一个高电平输入得到一个低电平输出。同时，Q_2 的集电极处在足够低的电压电平以使 Q_4 保持截止。

当输入为低电平时，Q_1的基极-发射极结是正向偏置，而基极-集电极结是反向偏置。电流通过R_1和Q_1的基极-发射极结导致低电平输入。低电平为电流接地提供了通路。Q_2的基极没有电流进入，因此Q_2截止。Q_2的集电极是高电平，从而使Q_4导通。饱和导通的Q_4提供了从V_{CC}到输出的低电阻通路，因此得到一个低电平输入对应一个高电平输出。同时，Q_2的发射极处在接地电位，保持Q_3截止。

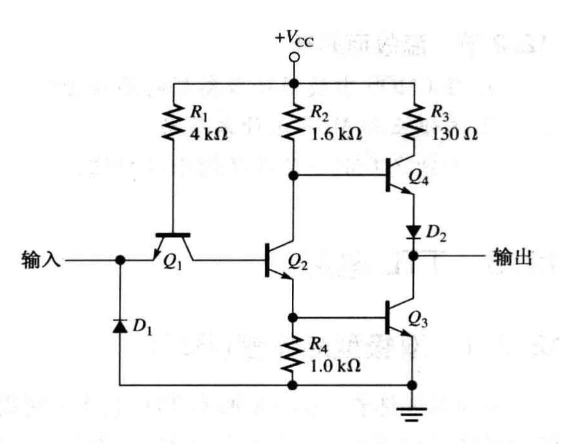


图 12.27 标准 TTL 反相器电路

TTL 电路中的二极管D_1防止输入负的尖峰电压，以避免损坏Q_1。二极管D_2确保Q_2导通(高电平输入)时，Q_4截止。在这种情况下，Q_2的集电极电压等于Q_3的基极-发射极电压V_{BE}加上Q_2的集电极-发射极电压V_{CE}。二极管D_2提供了附加的等效电压降V_{BE}，它与Q_4的基极-发射极电压串联，以保证Q_2导通时Q_4截止。

TTL 反相器在这两个输入状态下的工作情况如图 12.28 所示。在图 12.28(a)的电路中，Q_1的基极对地电压是 2.1 V，因此Q_2和Q_3导通。在图 12.28(b)的电路中，Q_1的基极对地电压是 0.7 V，不足以使Q_2和Q_3导通。

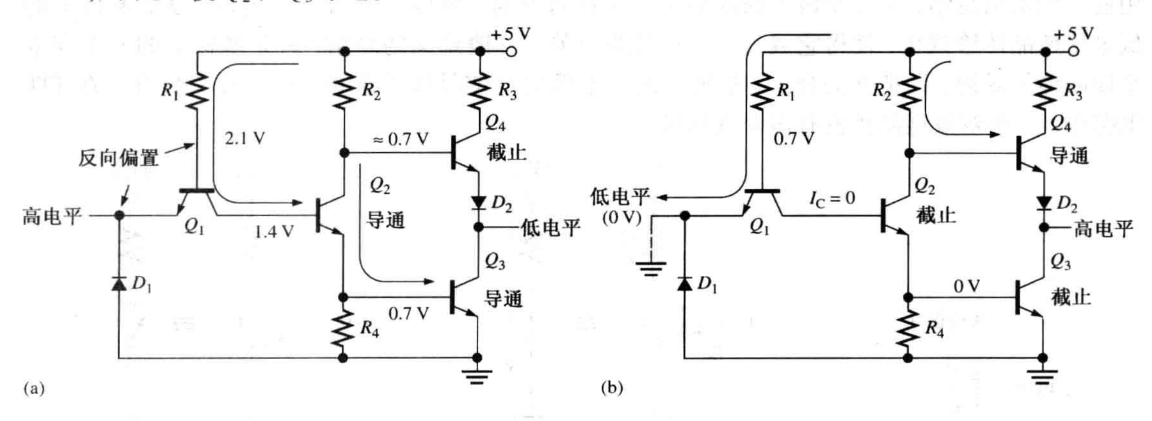


图 12.28 TTL 反相器的工作原理

12.3.3 TTL 与非门

如图 12.29 所示是一个 2 输入 TTL 与非门。它基本上与反相器电路相同，除了增加了Q_1的发射极输入。在 TTL 技术中，输入芯片使用了多发射极晶体管。这些多发射极晶体管可比做二极管的排列，如图 12.30 所示。

把图 12.29 的Q_1具体化，利用图 12.30 的二极管的连接替代Q_1，也许这样就可以更好地理解这个电路的工作原理。输入A或输入B上的低电平使各自的二极管正向偏置，使D_3(Q_1的基极-集电极结)反向偏置。这个反向偏置使Q_2保持截止，和所描述的 TTL 反相器相同，最后导致高电平输出。当然，两个输入同时为低电平也会产生相同的结果。

两个输入都是高电平，使得两个输入二极管都处在反向偏置，D_3(Q_1的基极-集电极结)正向偏置。这个正向偏置使Q_2导通，和所描述的 TTL 反相器相同，最后导致低电平输出。可以把以上的分析结果看成与非门的函数关系：当且仅当所有的输入都是高电平时，输出为低电平。

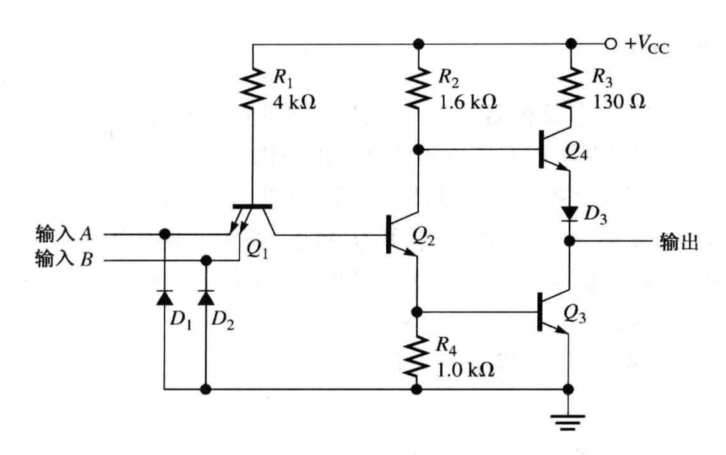


图 12.29　TTL 与非门电路

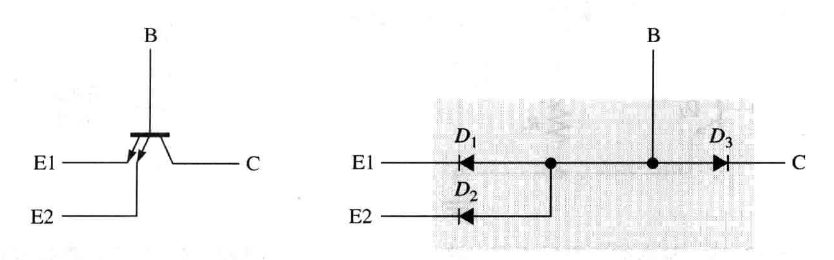


图 12.30　TTL 多发射极晶体管的二极管等效电路

12.3.4　集电极开路门

除了推拉输出电路，TTL 集成电路中可以使用的另一种类型的输出电路是集电极开路输出。它可以与 CMOS 的漏极开路输出相比较。具有集电极开路的标准的 TTL 反相器如图 12.31(a)所示。还有一些其他类型的门电路也具有集电极开路输出。

注意，晶体管 Q_3 的集电极是输出，没有连接任何东西，因此取名为“集电极开路”。为了在电路的输出得到正确的逻辑高电平和低电平，必须在 Q_3 的集电极到 V_{CC} 之间连接一个外部上拉电阻，如图 12.31(b)所示。当 Q_3 截止时，输出通过外部电阻被上拉到 V_{CC}。当 Q_3 导通时，输出通过饱和导通的晶体管连接到附近的地线。

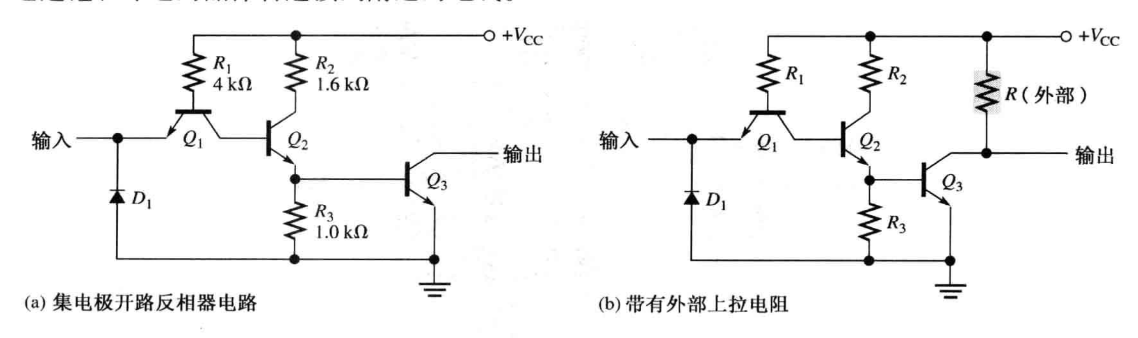


图 12.31　带有集电极开路输出的 TTL 反相器

指定反相器集电极开路输出的 ANSI/IEEE 标准符号如图 12.32 所示，与漏极开路输出相同。

12.3.5 TTL 三态门

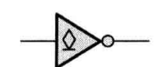


图 12.32 一个反相器中的集电极开路符号

图 12.33 给出了 TTL 三态反相器的基本电路。当使能输入为低电平时，Q_2截止，输出电路所起的作用为通常的推拉输出，输出状态依赖于输入状态。当使能输入为高电平时，Q_2导通。因此在Q_1的第二个发射极上是低电平，从而使得Q_3和Q_5截止，D_1是正向偏置，造成Q_4也截止。当两个推拉晶体管都截止时，它们在效果上是开路的，输出和内部电路完全断开，如图 12.34 所示。

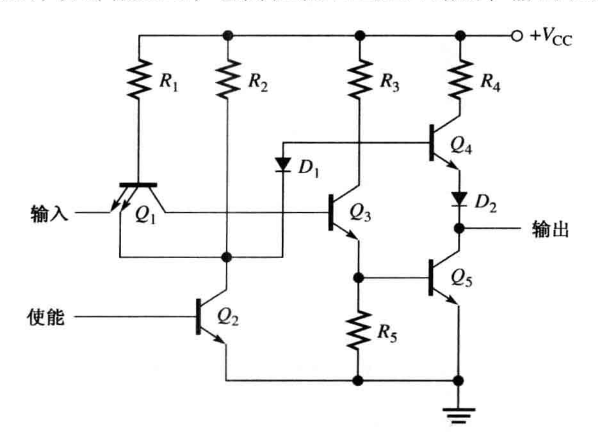


图 12.33 基本的三态反相器电路

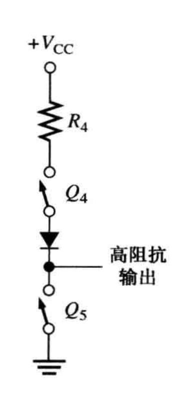


图 12.34 处于高阻抗状态的三态输出等价电路

12.3.6 肖特基 TTL

前面已经介绍了基本的或标准的 TTL 与非门电路。它是一种拉电流型的逻辑电路，当输出状态为低电平时，电路从负载上获取电流；当输出状态为高电平时，提供给负载的灌电流可以忽略。现在大多数 TTL 逻辑电路都使用某些形式的肖特基 TTL，通过采纳肖特基二极管以防止晶体管进入饱和状态，使得肖特基 TTL 具有更快的开关速度，因为减少了晶体管的开关时间。图 12.35 给出了一个肖特基门电路。请注意肖特基晶体管和肖特基二极管的符号。肖特基芯片以芯片编号中的“S”表示，如 74S00。其他一些类型的肖特基 TTL 电路，如低功耗的肖特基电路以 LS 表示，高级肖特基电路以 AS 表示，高级低功耗肖特基电路以 ALS 表示，快速肖特基电路以 F 表示。

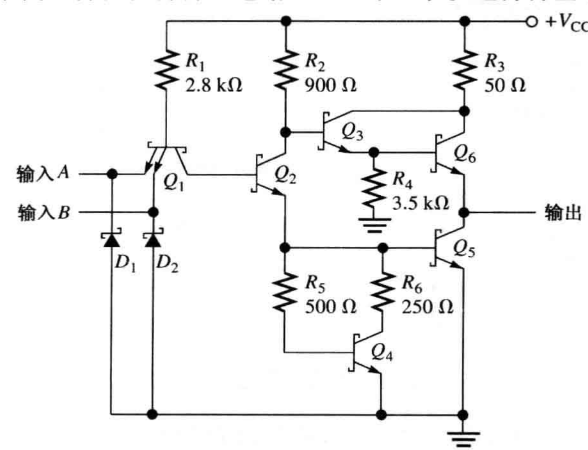


图 12.35 肖特基 TTL 与非门电路

12.3 节　温故而知新

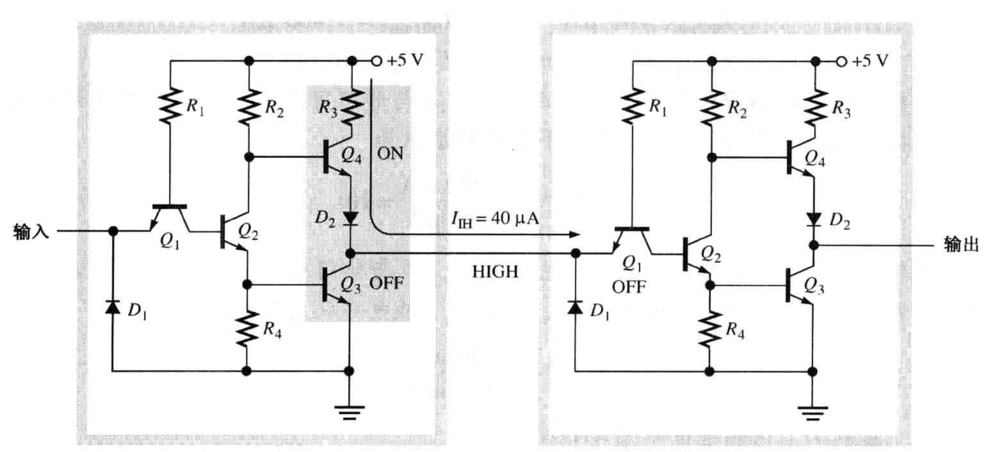
1. 对于一个 npn 双极型晶体管，当基极的电位比发射极低时，晶体管导通。(T 或 F)
2. 按开关的作用来定义，一个双极型晶体管的导通和截止状态表示什么？
3. 在 TTL 中输出电路的两种主要类型是什么？
4. 解释三态逻辑和通常的两种状态逻辑的区别。

12.4　TTL 在实际使用中的注意事项

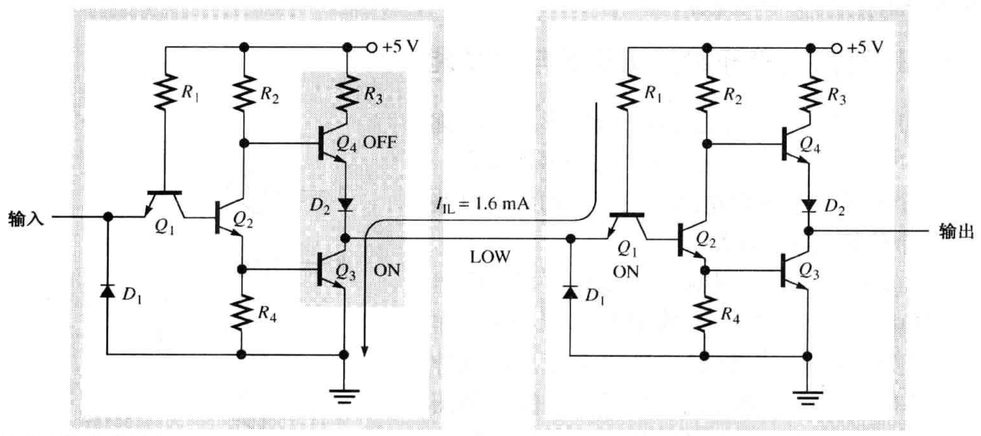
12.4.1　拉电流和灌电流

拉电流和灌电流的概念在 12.1 节已经介绍过。现在已经熟悉了在 TTL 中使用的推拉输出电路这样的结构，下面进一步了解拉电流和灌电流的形成过程。

图 12.36 给出了一个标准的 TTL 反柘器，它的推拉输出连接到另一个 TTL 反相器的输入。当驱动门电路处在高电平输出状态时，驱动器将向负载提供灌电流，如图 12.36(a)所示。对负载门电路的输入就像一个反向偏置的二极管，所以负载几乎不需要电流，事实上，因为输入是非理想的，所以从驱动器的推拉输出到负载门输入有 40 μA 的最大电流值。



(a)　灌电流(I_{IH}值最大)



(b) 拉电流 (I_{IL}值最大)

图 12.36　TTL 中的灌电流与拉电流

当驱动门电路处于低电平输出状态时，驱动器从负载获取拉电流，如图12.36(b)所示。对于标准TTL电路来说，这个电流的最大值为1.6 mA，由于超出了输入范围，所以在数据表中以负值表示。

例12.3 当TTL与非门驱动5个TTL输入时，那么驱动输出的灌电流是多少，拉电流是多少？(参见图12.36。)

解： 总的灌电流(高电平输出状态)是

$$I_{\mathrm{IH(max)}} = 40\ \mu\mathrm{A}/\text{输入}$$

$$I_{\mathrm{T(source)}} = (5\text{个输入})(40\ \mu\mathrm{A}/\text{输入}) = 5(40\ \mu\mathrm{A}) = \mathbf{200\ \mu A}$$

总的拉电流(低电平输出状态)是

$$I_{\mathrm{IL(max)}} = -1.6\ \mathrm{mA}(\text{每个输入})$$

$$I_{\mathrm{T(sink)}} = (5\text{个输入})(-1.6\ \mathrm{mA}/\text{输入}) = 5(-1.6\ \mathrm{mA}) = \mathbf{-8.0\ mA}$$

相关问题： 对一个LS TTL NAND门重新进行计算。参照可以得到的www.ti.com上的数据表。

例12.4 参照www.ti.com的数据表，确定7400与非门的扇出数。

解： 按照数据表，电流的参数如下：

$$I_{\mathrm{IH(max)}} = 40\ \mu\mathrm{A} \qquad I_{\mathrm{OH(max)}} = -400\ \mu\mathrm{A}$$

$$I_{\mathrm{IL(max)}} = -1.6\ \mathrm{mA} \qquad I_{\mathrm{OL(max)}} = 16\ \mathrm{mA}$$

对于高电平输出状态的扇出计算如下：电流$I_{\mathrm{OH(max)}}$是与非门可以提供给负载的最大电流。每个负载输入需要40 μA的电流$I_{\mathrm{IH(max)}}$。高电平状态的扇出为

$$\left|\frac{I_{\mathrm{OH(max)}}}{I_{\mathrm{IH(max)}}}\right| = \frac{400\ \mu\mathrm{A}}{40\ \mu\mathrm{A}} = \mathbf{10}$$

对于低电平输出状态，扇出的计算如下：电流$I_{\mathrm{OL(max)}}$是与非门可以获取的最大电流。

每个负载输入产生一个-1.6 mA的电流$I_{\mathrm{IL(max)}}$。低电平状态的扇出为

$$\left|\frac{I_{\mathrm{OL(max)}}}{I_{\mathrm{IL(max)}}}\right| = \frac{16\ \mathrm{mA}}{1.6\ \mathrm{mA}} = \mathbf{10}$$

在此情况下，高电平扇出和低电平扇出相同。

相关问题： 确定74LS00非门的扇出。

12.4.2 使用集电极开路门实现线与的功能

集电极开路门的输出可连接在一起，形成一种称为“线与”的电路。图12.37给出4个反相器连接在一起，产生一个4输入与非门电路。在所有的线与电路中，需要一个外部上拉电阻R_p。

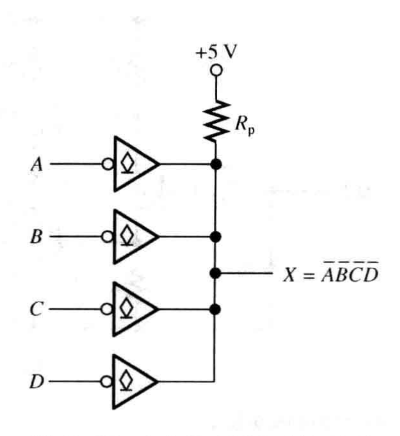


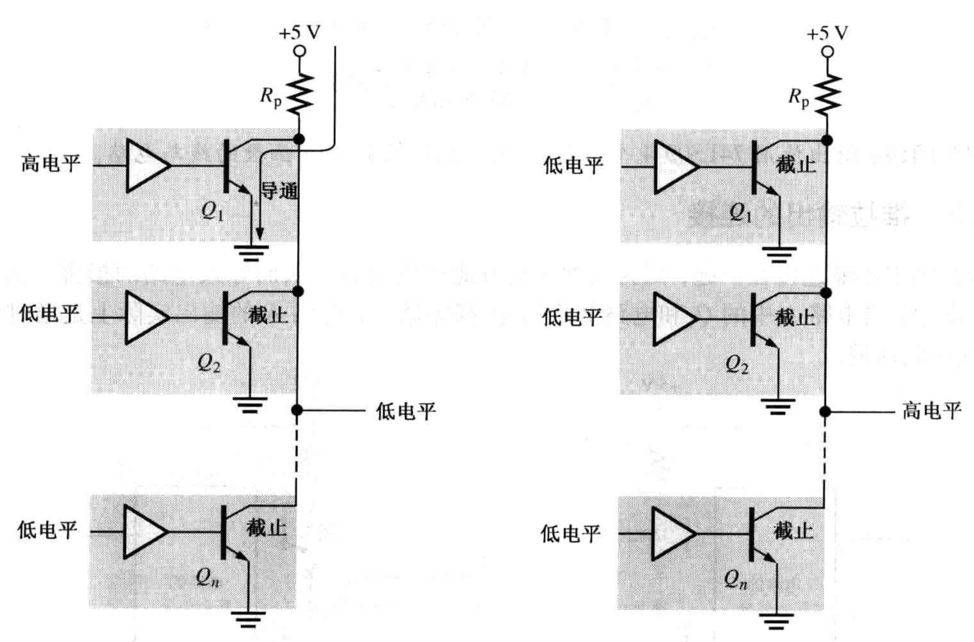
图12.37 4个反相器的线与连接

当一个(或多个)反相器输入为高电平时，输出X被拉为低电平，因为有一个输出晶体管导通，作为闭合的开关连接地，如图12.38(a)所示。在这种情况下，只有一个反相器为高电平输入，但是如图所示，通过饱和导通的输出晶体管，已经足够把输出Q_1拉为低电平。

对于 X 为高电平的情况，反相器所有的输入都必须为低电平，这样所有的集电极开路都截止，如图 12.38(b)所示。当这种情况发生时，输出 X 通过上拉电阻被拉到高电平。因此，当且仅当所有的输入为低电平时，输出 X 为高电平。所以，得到了一个与非函数，由下面的公式表示：



$$X = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$$



(a) 当一个或多个输出晶体管导通时，输出为低电平　　(b) 当所有输出晶体管截止时，输出为高电平

图 12.38　反相器集电极开路门的线与的工作原理

例 12.5　写出图 12.39 所示的集电极开路与门的线与电路的输出表达式。

解： 输出表达式为

$$X = ABCDEFGH$$

4 个 2 输入与门电路的线与创建了一个 8 输入的与门电路。

相关问题： 如果在图 12.39 中使用与非门电路，确定输出表达式。

例 12.6　3 个集电极开路与门连接成线与电路，如图 12.40 所示。假定此线与电路驱动 4 个标准 TTL 输入(每个为 −1.6 mA)。

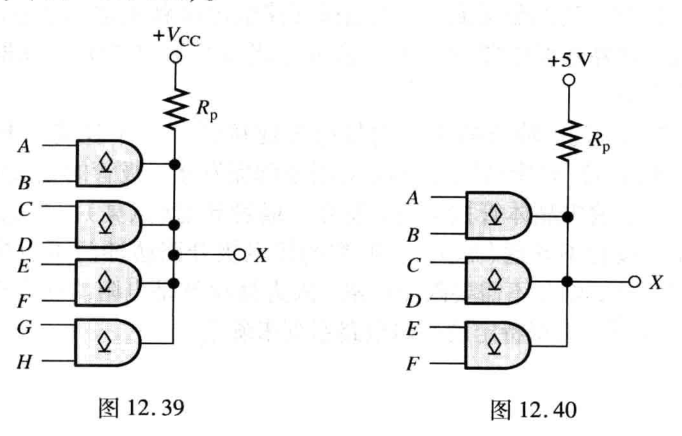


图 12.39　　图 12.40

(a)写出 X 的逻辑表达式；

(b)如果每个门的 $I_{OL(max)}$ 是 30 mA，$V_{OL(max)}$ 是 0.4 V，确定 R_p 的最小值。

解：(a) $X = ABCDEF$

(b) $4(1.6\ \text{mA}) = 6.4\ \text{mA}$

$$I_{R_p} = I_{OL(max)} - 6.4\ \text{mA} = 30\ \text{mA} - 6.4\ \text{mA} = 23.6\ \text{mA}$$

$$R_p = \frac{V_{CC} - V_{OL(max)}}{I_{R_p}} = \frac{5\ \text{V} - 0.4\ \text{V}}{23.6\ \text{mA}} = \mathbf{195\ \Omega}$$

相关问题：给出使用 74LS09 4 个 2 输入与门组成 10 输入与函数的线与电路。

12.4.3 推拉输出的连接

推拉输出不能连接在一起，因为这种连接可能产生过电流从而导致电路的损害。例如，在图 12.41 中，当电路 A 中的 Q_1 和电路 B 中的 Q_2 都导通时，电路 A 的输出实际上是通过电路 B 的 Q_2 短接到地的。

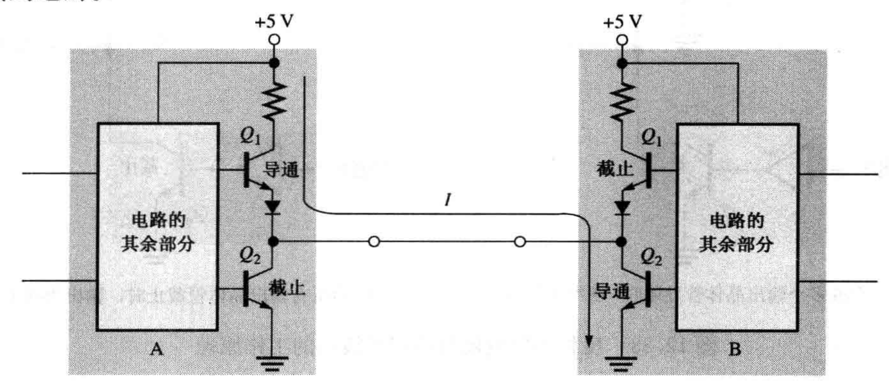


图 12.41 推拉输出线与在一起。这样的连接可能导致过电流经过电路 A 中的 Q_1 和电路 B 中的 Q_2，永远不应该这样使用

12.4.4 集电极开路缓冲器/驱动器

带有推拉输出的 TTL 电路的电流是有限制的，对于标准的 TTL 电路，在低电平状态下，它的拉电流($I_{OL(max)}$)为 16 mA，对于 LS TTL 电路则为 8 mA。在许多特殊应用中，一个门电路必须驱动外部电路，如发光二极管(LED)、电灯或者继电器，这就需要更多的电流。

由于其更高的电压和电流的驱动能力，集电极开路输出电路通常用于驱动发光二极管、电灯或者继电器。但是，只要外部电路所需要的输出电流没有超出 TTL 驱动器能够提供的拉电流，就可以使用推拉输出。

使用集电极开路 TTL 门，输出晶体管的集电极连接到了一个发光二极管或白炽灯，如图 12.42 所示。在图 12.42(a)中的限流电阻 R_L 用于确保发光二极管的电流低于最大值。当门电路的输出为低电平时，输出晶体管拉电流，发光二极管导通(点亮)。当输出晶体管截止时，输出是高电平，发光二极管不导通(熄灭)。典型的集电极开路缓冲门可以提供的拉电流高达 40 mA。在图 12.42(b)中，灯并不需要限流电阻，因为灯丝就是电阻。通常在集电极开路处可以使用高达 +30 V 的电压，根据特定的逻辑电路系列来确定。

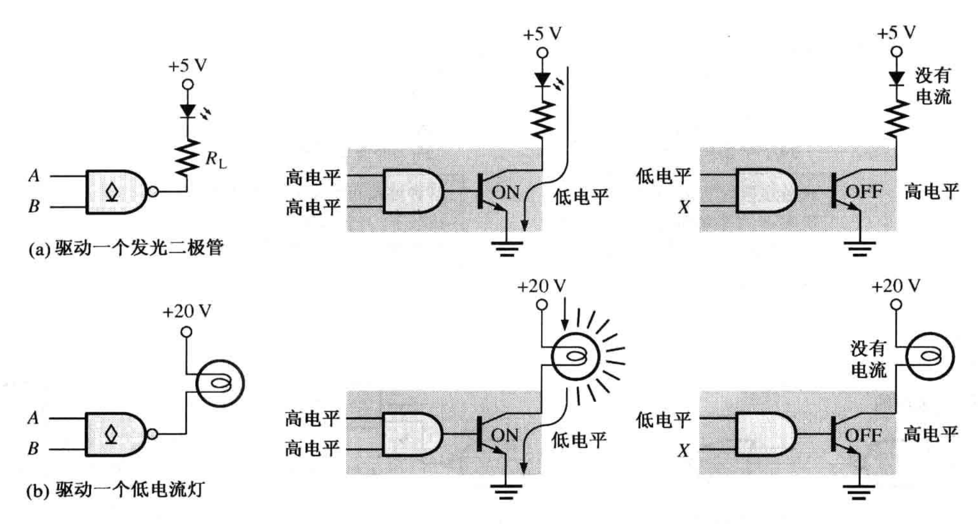


图 12.42　集电极开路驱动器的一些应用

例 12.7　确定图 12.43 的集电极开路电路的限流电阻 R_L 的阻值，假设发光二极管的电流为 20 mA，正向偏置的电压降为 1.5 V。门电路输出电压在低电平状态时为 0.1 V。

+5 V

R_L

A

B

图 12.43

解：

$$V_{R_L} = 5\text{ V} - 1.5\text{ V} - 0.1\text{ V} = 3.4\text{ V}$$

$$R_L = \frac{V_{R_L}}{I} = \frac{3.4\text{ V}}{20\text{ mA}} = \mathbf{170\ \Omega}$$

相关问题： 如果发光二极管需要 35 mA 的电流，确定限流电阻 R_L 的阻值。

12.4.5　未用的 TTL 输入

TTL 门中悬空的输入是作为高电平，因为悬空的输入导致输入晶体管的发射极结反向偏置，就如同高电平的效果。这种效应如图 12.44 所示。但是，由于对噪声的敏感，所以最好不要使未用的 TTL 输入悬空（开路）。这里有几种可选择的方法用于处理未用的输入。

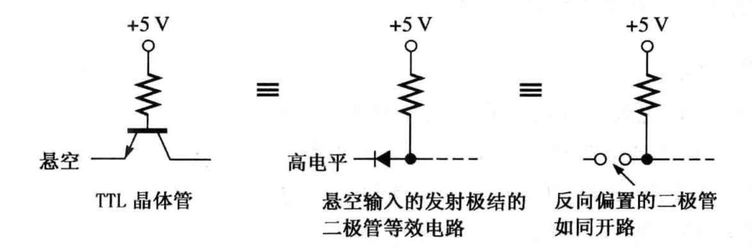


图 12.44　开路的 TTL 输入和高电平输入的比较

输入连接在一起　处理未使用门电路输入的最常用的方法，就是将它们连接到同一个门的一个使用的输入上。对于与门和与非门，在低电平状态下，所有连接在一起的输入看成一个单位负载；但对于或门和或非门来说，在低电平状态下，每个连到其他输入的输入都看成为各自的单位负载。在高电平状态下，对于所有类型的 TTL 门电路，连接在一起的每个输入都看成为各自的负载。图 12.45(a)给出了两个例子，其中两个未用的输入连接到一个已使用的输入上。

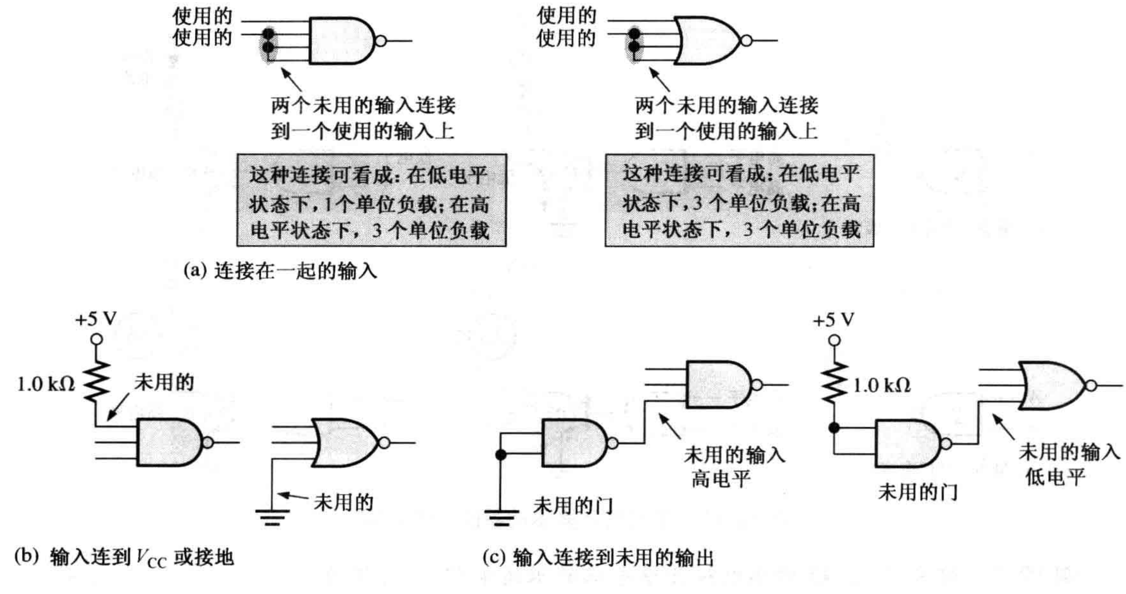


图 12.45　未用的 TTL 输入的处理方法

无论多少个输入连接在一起，与门和与非门都只提供一个单位负载，而对于或门和或非门，连接在一起的每个输入都是一个单位负载。这是因为与非门使用一个多发射极输入的晶体管；所以无论多少个输入是低电平，低电平状态的总电流都被限制为一个固定的数值。或非门电路对每个输入都使用各自的晶体管；因此，低电平状态的电流是所有连接在一起的输入电流的和。

输入连接到 V_{CC} 或接地　可以把与门和与非门的未用的输入通过一个 1.0 kΩ 的电阻连接到 V_{CC}。这样连接可使未用的输入拉到高电平。或门和或非门的未用的输入可以连接到地。这些方法如图 12.45(b)所示。

输入连接到未用的输入　第三种处理未用的输入的方法，可用于存在一个未用的门或反相器的情况下。对于未用的与门和与非门，输出必须始终是高电平；对于不使用的或门和或非门，输出必须始终是低电平，如图 12.45(c)所示。

12.4 节　温故而知新

1. TTL 电路从负载流入电流时，其输出状态是什么？
2. 为什么 TTL 电路对于负载的灌电流比拉电流要小？
3. 为什么具有推拉输出的 TTL 电路不能连接在一起？
4. 什么类型的 TTL 电路必须使用线与来连接？
5. 什么类型的 TTL 电路可以用来驱动一个灯？
6. 一个未用的 TTL 输入处于低电平。(T 或 F)

12.5　CMOS 和 TTL 性能的比较

过去，与 CMOS 电路相比，TTL(双极型)的出众特性表现在它相对较高的速度和输出电流能力方面。虽然 TTL 仍然还在使用，但是 CMOS 在许多方面通常与 TTL 相当或比 TTL 更为出

众，在这一点上使得 TTL 的使用已经减少，而 CMOS 已成为了主流的 IC 技术。IC 逻辑芯片的一个系列——BiCMOS，结合了 CMOS 逻辑与 TTL 输出的电路，吸取了两者的优点。

表 12.1 提供了几种 IC 逻辑系列工作性能的比较。

表 12.1　比较几种 74XXIC 系列的性能参数

	双极型 (TTL)			BiCMOS	CMOS					
					5 V			3.3 V		
	F	LS	ALS	ABT	HC	AC	AHC	LV	LVC	ALVC
速度										
门传输延迟，t_p (ns)	3.3	10	7	3.2	7	5	3.7	9	4.3	3
FF 最大时钟频率 (MHz)	145	33	45	150	50	160	170	90	100	150
每个门的功耗										
双极型：50% dc (mW)	6	2.2	1.4							
CMOS：静态 (μW)				17	2.75	0.55	2.75	1.6	0.8	0.8
输出驱动										
I_{OL} (mA)	20	8	8	64	4	24	8	12	24	24

12.5 节　温故而知新

1. BiCMOS 电路是什么？
2. 一般情况下，CMOS 和双极型(TTL)相比的主要优点是什么？

12.6　发射极耦合逻辑(ECL)电路

ECL 或门/或非门(OR/NOR)电路如图 12.46(a)所示。射极跟随器输出电路提供了或逻辑功能和或非的补码，如图 12.46(b)所示。

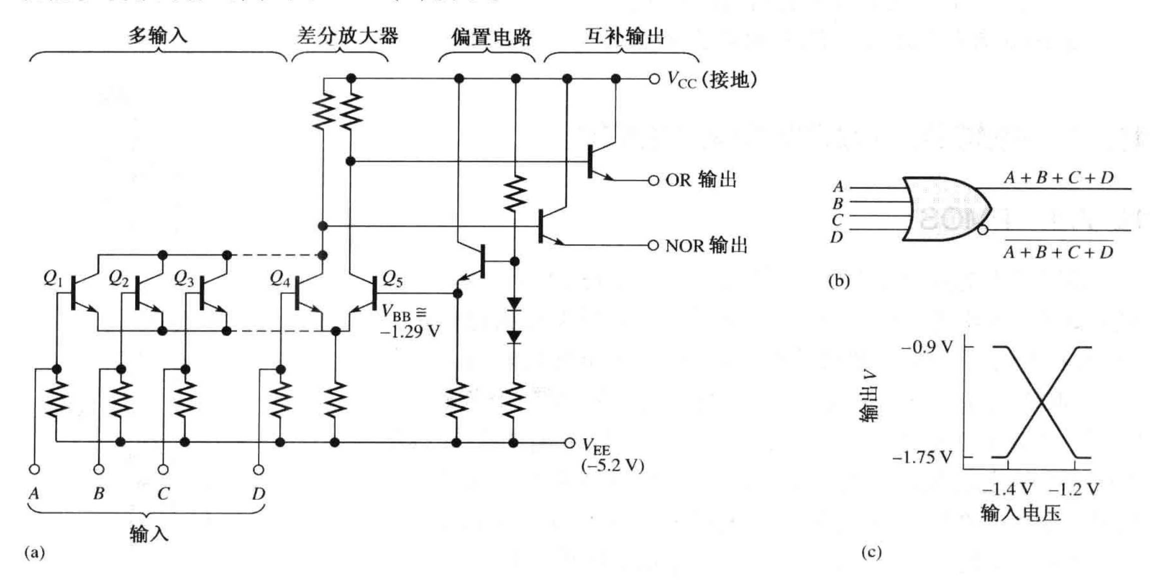


图 12.46　一个 ECL OR/NOR 门电路

因为发射极跟随器的输出阻抗较低，而差分放大器输入的输入阻抗较高，所以高扇出的驱动是可能的。在这种类型的电路中，不可能出现饱和状态。由于没有饱和状态，导致较高的功耗和受到限制的电压波动(小于1 V)，但允许高频开关工作。

V_{CC}引脚通常是连接到地，V_{EE}引脚连接电源电压 -5.2 V，以便处于最佳工作状态。注意图12.46(c)中的输出对地电平从低电平 -1.75 V 到高电平 -0.9 V 变化。在正逻辑中，1是高电平(比负值小)，0是低电平(更小的负值)。

12.6.1 噪声容限

正如所学到的，门电路的噪声容限是衡量对不需要的电压波动(噪声)的抗干扰能力。典型的ECL电路的噪声容限大约从0.2 V到0.25 V。这两个数字都小于TTL的指标，从而使得ECL不太适合噪声大的环境。

12.6.2 ECL和TTL与CMOS的比较

表12.2给出了对F、AHC和ECL的关键性能参数的比较。

表12.2 ECL系列的特性参数与F和AHC的特性参数的比较

	双极型(TTL) F	CMOS AHC	双极型(ECL)
速度			
门传输延迟，t_p (ns)	3.3	3.7	0.22~1
FF最大时钟频率(MHz)	145	170	330~2800
每个门的功耗			
双极型：50% dc	6 mW		25~73 mW
CMOS：静态		2.75 μW	

12.6节 温故而知新

1. ECL与TTL相比的主要优点是什么？
2. 给出两个ECL与TTL相比的优点。

12.7 PMOS、NMOS和E^2CMOS

12.7.1 PMOS

最初产生的高密度MOS电路技术之一就是PMOS。它使用增强模式的p沟道MOS晶体管构成了基本的门电路构建模块。图12.47给出了产生正逻辑或非功能的基本PMOS门电路。

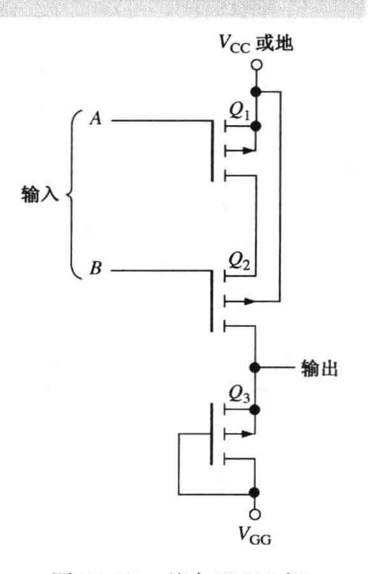


图12.47 基本PMOS门

PMOS门电路的工作原理如下：电源电压V_{GG}是一个负电压，V_{CC}是一个正电压或地(0 V)。晶体管Q_3一直处于偏置状态，以生成一个恒定的漏源电阻。它的唯一目的就是作为限流电阻。如果一个高电平(V_{CC})加在输入A或输入B上，那么Q_1或Q_2截止，输出被拉到一个接近V_{GG}的电压，表示低电平。如果一个低电平电压(V_{GG})加在输入A和输入B上，那么Q_1和Q_2导通。造成输出变为高电平(接近V_{CC})，因为两个输入中只要有一个是高

电平时，输出就是低电平，而只有所有的输入都是低电平时，输出才是高电平，所以得到了一个或非门。

12.7.2 NMOS

NMOS 芯片是作为改进的处理技术开发的。NMOS 电路使用 n 沟道 MOS 晶体管，如图 12.48 中的与非门和或非门电路所示。

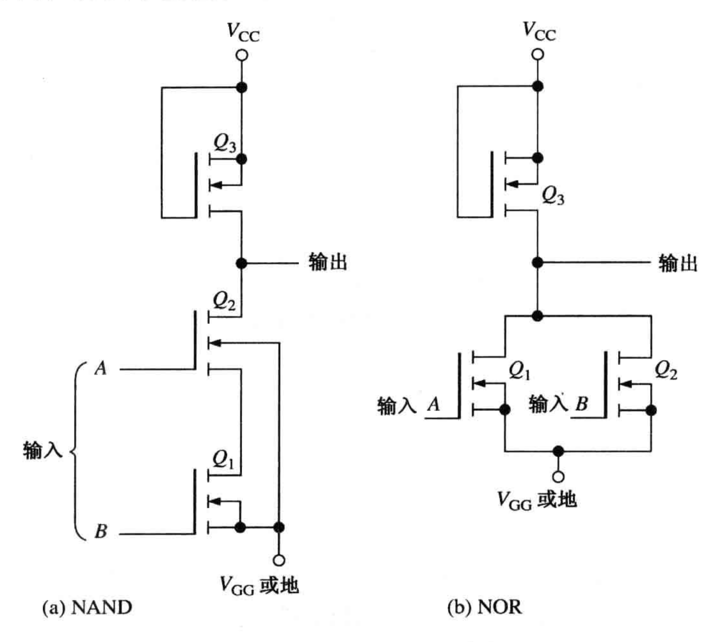


图 12.48　两个 NMOS 门

在图 12.48(a)中，Q_3作为一个限流电阻。当一个低电平(V_{GG}或地)加到一个输入或两个输入上时，那么至少有一个晶体管(Q_1或 Q_2)截止，输出被拉到高电平，接近 V_{CC}。当高电平加到输入 A 和 B 上时，Q_1和 Q_2都导通，输出为低电平。当然，这个分析结果得出此电路是一个与非门电路。

在图 12.48(b)中，Q_3再次作为一个电阻。两个中有一个输入是高电平，就会使 Q_1或 Q_2导通，并将输出拉到低电平。当两个输入都是低电平时，两个晶体管都截止，输出被拉到高电平。

12.7.3 E^2CMOS

E^2CMOS(电可擦除)技术基于 CMOS 和 NMOS 技术的结合，用在可编程逻辑芯片中，例如 ROM 和 CPLD。一个 E^2CMOS 单元是围绕带浮动门的 MOS 晶体管构建的，这种浮动门由一个不大的编程电流进行外部充电和放电。这种类型的单元示意图如图 12.49 所示。

当浮动门通过移出电子充电到正电压时，读取晶体管导通，并存储二进制 0。当浮动门电路通过放入电子充电到负电压时，读取晶体管截止，并存储二进制 1。控制门控制浮动门的电压。在使用字和位线的读写操作的过程中，传递晶体管把读取晶体管与字和位线阵列隔离开来。

E^2CMOS 单元可通过在控制门或单元的位线加一个编程脉冲进行编程，而位线已由字线上的电压选中。在编程周期中，首先通过在控制门上加一个电压使得浮动门为负电压，这样

E^2CMOS单元的内容被擦除。这将使读取晶体管处于截止状态(存储一个1)。在单元的位线加一个写脉冲将保存一个0。这将使浮动门充电到一个使读取晶体管导通(存储0)的电压。通过读取位线上是否存在一个小的单元电流，单元中存储的位就被读出。如果存储的是1，那么就没有单元电流，因为读取晶体管截止。如果存储的是0，则有一个小的单元电流，因为读取晶体管导通。一旦在单元中存储了一位，就会永远保存，除非擦除这个单元或者在这个单元中写入了一个新的位。

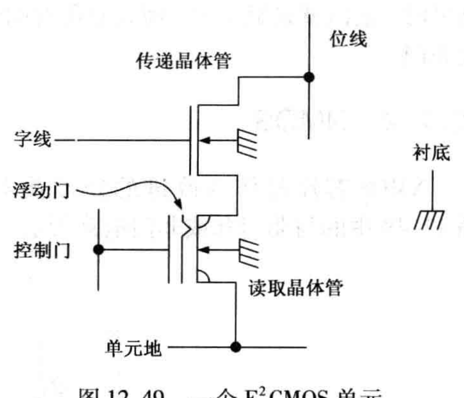


图 12.49 一个 E^2CMOS 单元

12.7 节 温故而知新

1. 在集成电路中，NMOS 和 PMOS 技术的主要特点是什么？
2. 在一个 E^2CMOS 单元中，实现电荷存储的机械装置是什么？

关键词

CMOS 互补金属氧化物半导体；一种集成逻辑电路，使用n沟道和p沟道 MOSFET(互补金属氧化物半导体场效应管)。

拉电流 一个逻辑电路从负载接收电流到输出端的行为。

灌电流 一个逻辑电路从输出端送出电流到负载的行为。

ECL 射极-耦合逻辑；集成电路的一种，由不饱和双极型晶体管形成。

E^2CMOS 电可擦除 CMOS，使用可编程逻辑设备(PLD)的 IC(集成电路)技术。

扇出 一个逻辑门可以驱动的相同系列的等效门输入的数目。

抗噪声 一个逻辑电路拒绝不需要的信号的能力。

噪声容限 一个门的最大低电平输出和一个同等的门的最大低电平输入之间的差，也可以是最小高电平输出和一个同等的门的最小高电平输入之间的差。噪声容限有时用直流电源电压的百分比来表示。

集电极开路 TTL电路的一种，输出晶体管的集电极内部没有连接，可以从外部连接到需要大电流和电压的负载上。

功率损耗(功耗) 在一个电子电路中，直流电源电压和电流的乘积。

传输延迟时间 在一个逻辑电路中，输入的变化到对应的输出变化之间的时间间隔。

上拉电阻 一个电阻的一端连接到使用的直流电源，当逻辑电路处在无效状态时，连接电路的电阻的另一端就保持高电平。

推拉(图腾) TTL电路的一种输出。

三态 逻辑电路输出的一种类型，三种状态有高电平、低电平和高阻抗。

TTL 晶体管-晶体管逻辑；集成电路的一种，使用双极结型晶体管，也称为双极型。

单位负载 扇出的一种测量单位。一个门输入表示驱动门的一个单位负载。

判断题 （答案在本章的结尾。）

1. TTL 的典型的直流电源电压是 +5 V。
2. 一个逻辑门的扇出是在 IC 封装中的门的数目。
3. CMOS 使用 MOSFET。
4. BJT 表示双极型晶体管。
5. 一个集电极开路门必须连接一个外部电阻。
6. CMOS 主导了数字 IC(集成电路)技术。
7. 一个推拉输出意思是两个或多个电阻串联。
8. CMOS 受 ESD(静电放电)的限制。
9. 一个三态输出可以是高电平、低电平或高阻抗。
10. 传输延迟是一个逻辑门速度的测量。

自测题 （答案在本章的结尾。）

1. 如果 CMOS 门电路输入信号的频率增加，那么平均的功耗将
   (a)减少 (b)增加 (c)不会变化 (d)呈指数减少
2. 在高噪声的环境中，CMOS 的工作比 TTL 更可靠，这是因为 CMOS 的
   (a)较低的噪声容限 (b)输入电容
   (c)较高的噪声容限 (d)较小的功耗
3. 以适当的方法处理 CMOS 芯片是非常必要的，这是因为其
   (a)脆弱的结构 (b)高噪声容限
   (c)对静电放电的敏感性 (d)低功耗
4. 下面哪一个不是 TTL 电路？
   (a)74F00 (b)74AS00 (c)74HC00 (d)74ALS00
5. 一个悬空的 TTL 或非门输入端
   (a)作为低电平 (b)作为高电平
   (c)应该接地 (d)应当通过电阻连接到 V_{CC}
   (e)答案(b)和(c) (f)答案(a)和(c)
6. 一个 LS TTL 门电路能够驱动最多
   (a)20 个单元负载 (b)10 个单元负载 (c)40 个单元负载 (d)无限多个单元负载
7. 如果 LS TTL 门电路的两个不使用的输入连接到了由另一个 LS TTL 门电路驱动的输入上，那么剩下可以由这个门电路驱动的单元负载总数是
   (a)7 个 (b)8 个 (c)17 个 (d)没有限制
8. ECL 相对于 TTL 或 CMOS 电路的主要优点是
   (a)ECL 更便宜 (b)ECL 的功耗更低
   (c)ECL 可用在更多种类的电路中 (d)ECL 的速度更快
9. ECL 不能用在
   (a)高噪声的环境中 (b)潮湿的环境中 (c)高频应用中
10. E^2CMOS 单元中存储一个数据位的基本结构是
   (a)控制门电路 (b)浮动漏极 (c)浮动门 (d)单元电流

习题 (奇数题的答案在本书的结尾。)

12.1 节 基本操作特性和参数

1. 一个逻辑门电路的 $V_{OH(min)}=2.2$ V，它正在驱动一个 $V_{IH(min)}=2.5$ V 的门电路。这两个电路在高电平状态下工作是否适合？为什么？
2. 一个逻辑电路的 $V_{OL(max)}=0.45$ V，它正在驱动一个 $V_{IL(max)}=0.75$ V 的门电路。这两个电路在低电平状态下工作是否适合？为什么？
3. 一个 TTL 门电路具有下列实际的电压电平值：$V_{IH(min)}=2.25$ V，$V_{IL(max)}=0.65$ V。假设它正在由一个 $V_{OH(min)}=2.4$ V 和 $V_{OL(max)}=0.4$ V 的门电路驱动。那么高电平和低电平状态下的噪声容限是多少？
4. 对于习题 3 中的门电路，在高电平和低电平状态下，它所能允许的输入上的噪声尖峰电压的最大振幅是多少？
5. 表 12.3 中给出了 3 种类型的逻辑门的电压规范。请选择可在高噪声工业环境中使用的门电路。

表 12.3

	$V_{OH(MIN)}$	$V_{OL(MAX)}$	$V_{IH(MIN)}$	$V_{IL(MAX)}$
门 *A*	2.4 V	0.4 V	2 V	0.8 V
门 *B*	3.5 V	0.2 V	2.5 V	0.6 V
门 *C*	4.2 V	0.2 V	3.2 V	0.8 V

6. 某个门电路从 +5 V 的电源获得直流电流，在低电平状态下电流为 2 mA，在高电平状态下电流为 3.5 mA。在低电平状态下的功耗是多少？高电平状态下的功耗是多少？假定占空比为 50%，平均功耗又是多少？
7. 图 12.50 的电路中的每个门的 t_{PLH} 和 t_{PHL} 都是 4 ns。如图所示，如果在输入加一个正脉冲，那么经过多长的时间才能出现输出？

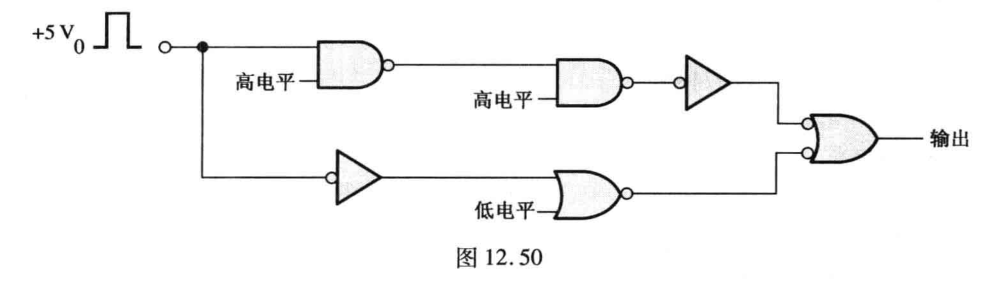


图 12.50

8. 给定一个门电路，$t_{PLH}=3$ ns 和 $t_{PHL}=2$ ns。平均传输延迟时间是多少？
9. 表 12.4 列出了 3 种类型的门电路的参数。根据对速度-功率乘积的判断，哪一个具有最佳的性能？

表 12.4

	t_{PLH}	t_{PHL}	P_D
门 *A*	1 ns	1.2 ns	15 mW
门 *B*	5 ns	4 ns	8 mW
门 *C*	10 ns	10 ns	0.5 mW

10. 如果希望门电路工作在可以达到的最高频率，选择表 12.4 中的哪一个门电路？
11. 标准的 TTL 门电路的扇出系数是 10。图 12.51 中是否有门电路超载？如果有，是哪一个？
12. 图 12.52 中的哪一个 CMOS 门电路网络的可以在最高频率下工作？

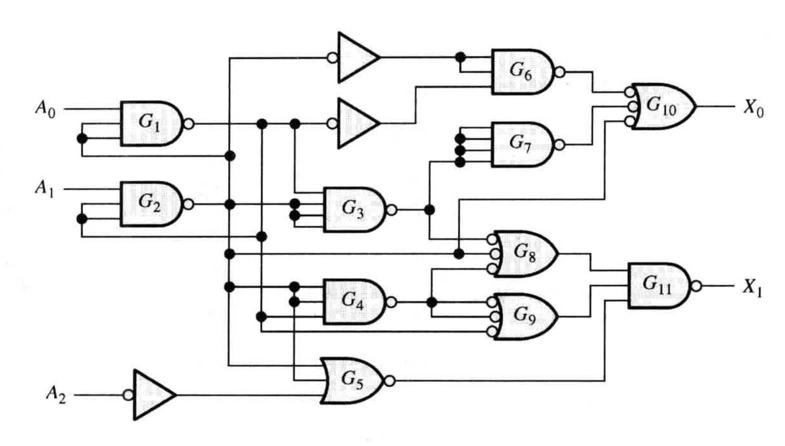


图 12.51

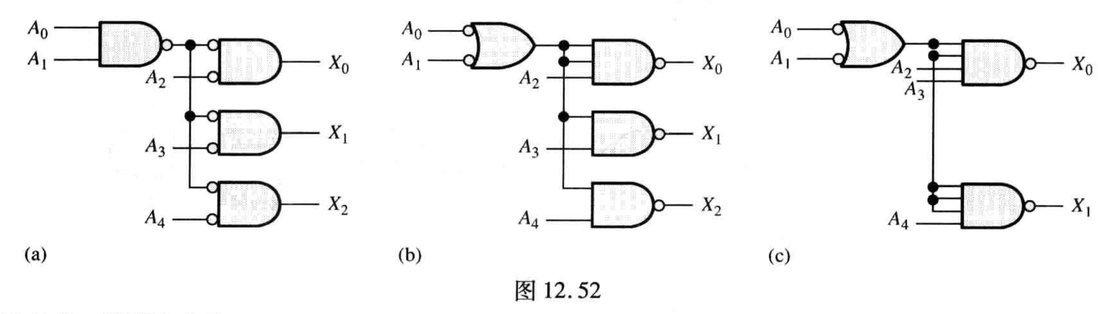


图 12.52

12.2 节　CMOS 电路

13. 确定图 12.53 中每一个 MOS 场效应管的状态(导通或截止)。

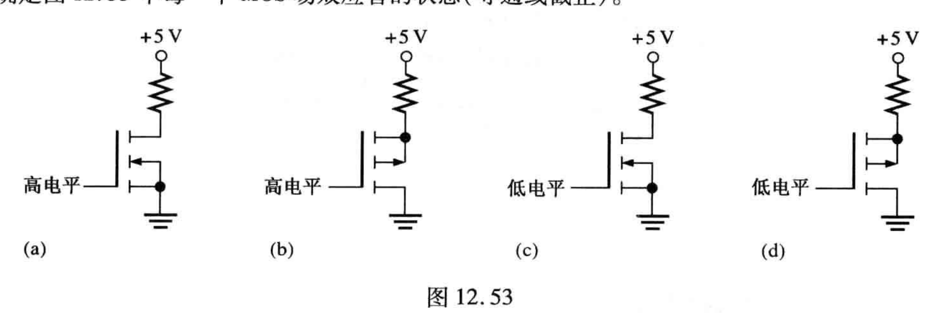


图 12.53

14. 图 12.54 中的 CMOS 门电路网络是不完整的。指出应该怎样修改。

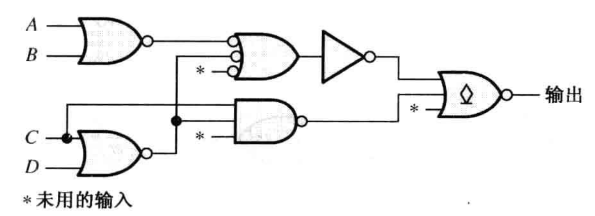


图 12.54

15. 使用合适的 CMOS 逻辑门电路和/或反相器设计一个电路，通过这个电路，来自 4 个不同源头的信号在不同的时间可连接到一个公共的线路上，并且不会互相干扰。

12.3节 TTL电路

16. 确定图12.55中哪一个BJT截止，哪一个导通。

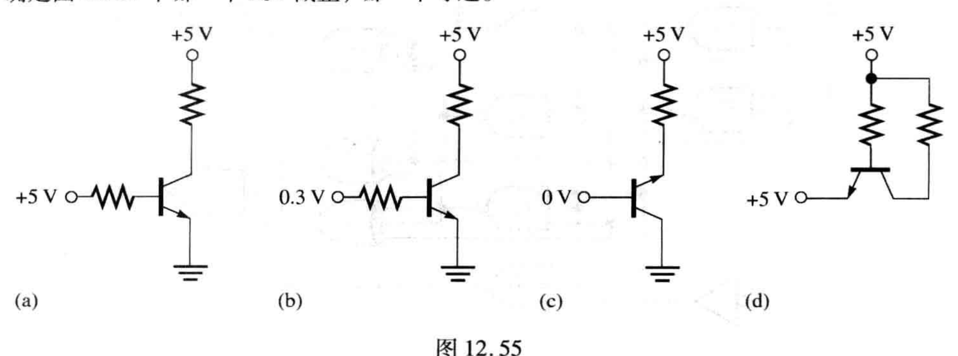


图12.55

17. 确定图12.56中每个TTL门电路的输出状态。

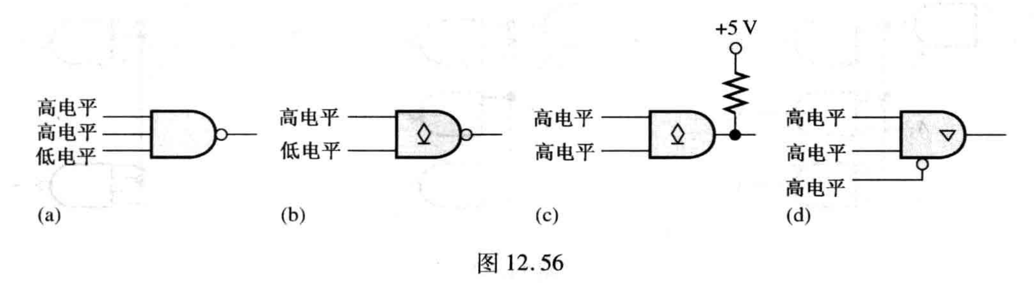


图12.56

18. 图12.57中的门电路网络是不完整的，指出应当怎样修改。

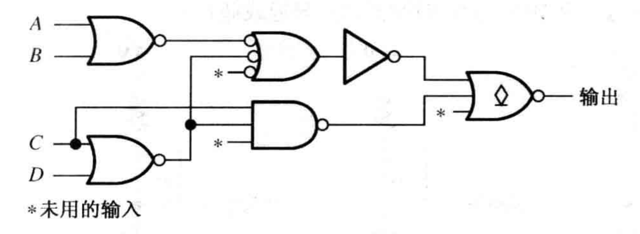


图12.57

12.4节 TTL在实际使用中的注意事项

19. 确定图12.58中每个TTL门电路的输出电平。

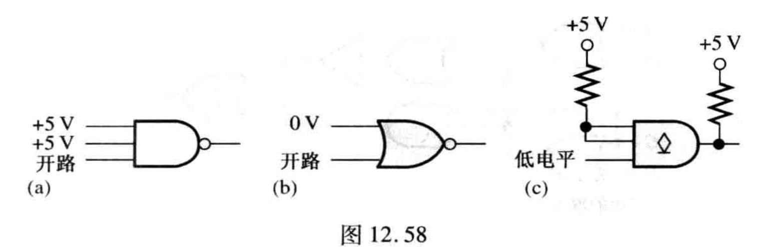


图12.58

20. 针对图12.59中的每个部分，说出每一个驱动门电路是灌电流还是拉电流。请详细说明每种情况下驱动门电路流进或流出的最大电流。所有的门电路都是标准TTL。

21. 使用集电极开路反相器实现下列逻辑表达式：

(a) $X = \overline{ABC}$　　(b) $X = A\overline{BCD}$　　(c) $X = ABC\overline{DEF}$

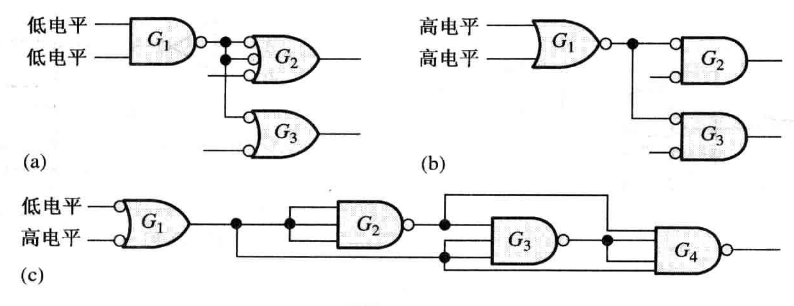


图 12.59

22. 写出图 12.60 中每个电路的逻辑表达式。

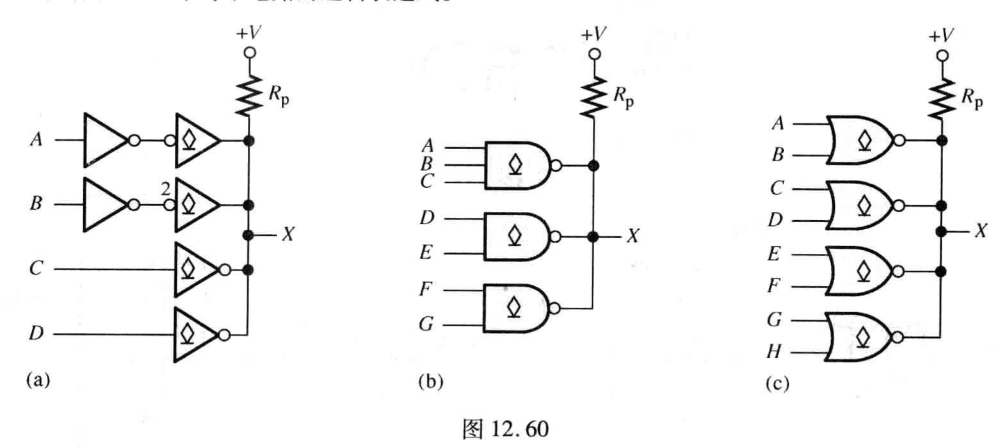


图 12.60

23. 如果对每个门电路 $I_{OL(max)}=40$ mA 和 $V_{OL(max)}=0.25$ V，确定图 12.60 中每个上拉电阻的最小值。假设输出 X 驱动了 10 个标准 TTL 单位负载，电源电压是 5 V。

24. 一个给定的继电器需要 60 mA 的电流。请使用集电极开路与非门电路驱动这个中继器，集电极开路与非门的 $I_{OL(max)}=40$ mA。

12.5 节　CMOS 和 TTL 性能的比较

25. 请在表 12.1 中选出速度-功率乘积最好的 IC 系列。

26. 请在表 12.1 的逻辑电路系列中选择最适合下列每种需要的逻辑电路：
   (a)传输延迟时间最短
   (b)触发器触发速度最快
   (c)功耗最低
   (d)逻辑门电路的速度和功耗之间的最佳折中

27. 确定图 12.61 的每个电路从每个输入到每个输出的总的传播延迟时间。

28. 图 12.62 的触发器中有一个可能有不稳定的输出。如果有，是哪一个，为什么？

12.6 节　发射极耦合逻辑(ECL)电路

29. ECL 逻辑电路系统和 TTL 逻辑电路系统之间有哪些基本的差别？

30. 针对下列每一种需求，选择 ECL、CMOS 或合适的 TTL 系列：
   (a)速度最快
   (b)功耗最低
   (c)高速度和低功耗之间(速度-功率乘积)的最佳折中

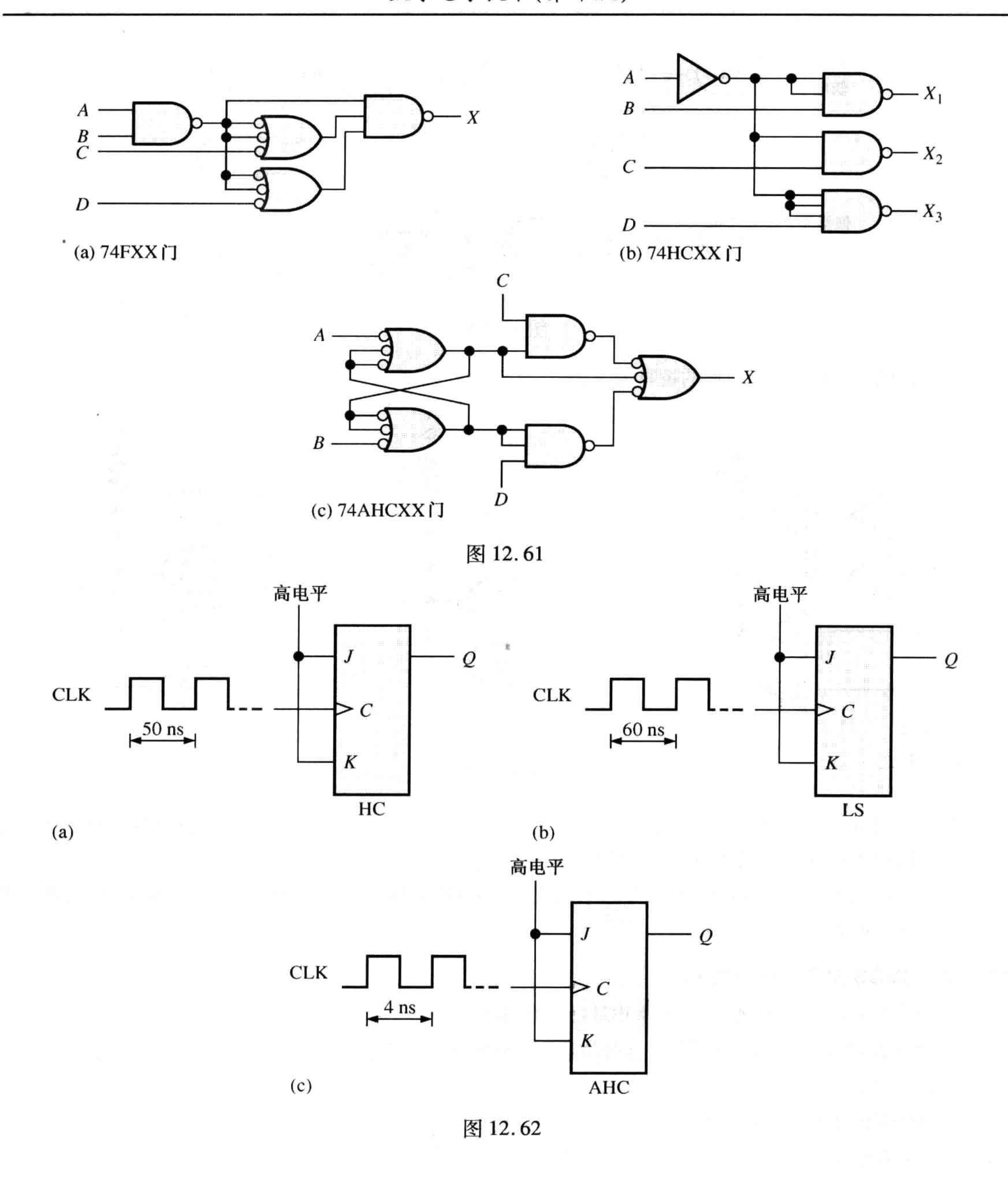


图 12.61

图 12.62

答案

温故而知新

12.1 节　基本操作特性和参数

1. V_{IH}：高电平输入电压；V_{IL}：低电平输入电压；V_{OH}：高电平输出电压；V_{OL}：低电平输出电压。
2. 一个更高值的噪声容限更好。
3. 门 B 可以运行在一个更高的频率下。
4. 过量的负载减少了一个门的噪声容限。

12.2 节　CMOS 电路

1. CMOS 逻辑中使用 MOSFET。
2. 一个电路的补输出由一个 n 沟道和 p 沟道 MOSFET 组成。
3. 因为静电放电可以损坏 CMOS 芯片。

12.3 节　TTL 电路

1. 错误，npn 双极型晶体管截止。
2. 双极型晶体管导通是一个闭合的开关，截止是一个打开的开关。
3. 推拉和集电极开路是 TTL 输出的不同类型。
4. 三态逻辑提供一个高阻抗状态，此时输出和其他的电路断开。

12.4 节　TTL 在实际使用中的注意事项

1. 在一个低电平状态下的拉电流。
2. 因为 TTL 负载看上去就像是一个在高电平状态下的反向偏置的二极管，所以灌电流比拉电流小。
3. 当一个输出为高电平，另一个是低电平时，推拉晶体管不能给出电流。
4. 线与必须使用集电极开路。
5. 灯驱动必须使用集电极开路。
6. 错误，一个未用的输入通常表现为高电平。

12.5 节　CMOS 和 TTL 性能的比较

1. BiCMOS 的输入和输出电路使用双极型晶体管，CMOS 介于中间。
2. CMOS 比双极型晶体管具有更低的功耗。

12.6 节　发射极耦合逻辑(ECL)电路

1. ECL 比 TTL 快。
2. ECL 比 TTL 功率大，噪声容限小。

12.7 节　PMOS、NMOS 和 E^2CMOS

1. NMOS 和 PMOS 是高密度的。
2. 浮动门是在 E^2CMOS 单元中保持电荷的机构。

例题的相关问题

12.1　CMOS。

12.2　10.75 μW。

12.3　$I_{T(source)} = 5(20\ \mu A) = 100\ \mu A$。

$I_{T(sink)} = 5(-0.4\ mA) = -2.0\ mA$。

12.4　扇出 = 20。

12.5　$X = (\overline{AB})(\overline{CD})(\overline{EF})(\overline{GH}) = (\overline{A}+\overline{B})(\overline{C}+\overline{D})(\overline{E}+\overline{F})(\overline{G}+\overline{H})$。

12.6　参见图 12.63。

12.7　$R_L = 97\ \Omega$。

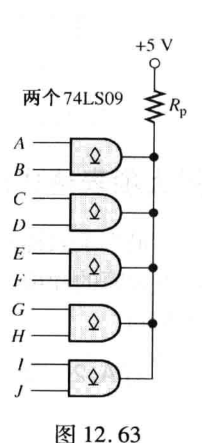


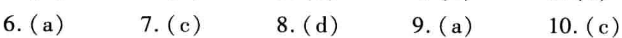
图 12.63

判断题答案

1.T　2.F　3.T　4.F　5.T　6.T　7.F　8.T　9.T　10.T

自测题答案

1.(b)　2.(c)　3.(c)　4.(c)　5.(e)



6.(a)　7.(c)　8.(d)　9.(a)　10.(c)

附录 A　卡诺图或与项(POS)的最小化

写出最大项(标准或与项)的表达式

对于标准形式的或与项，表达式中的每一个和项在对应卡诺图中放一个0。每个0放在与和项相对应的小方格中。例如，对于和项 $A+\overline{B}+C$，在3变量的卡诺图中对应的010小方格里放置一个0。

当最大项表达式(标准或与项表达式)的和项全部映射到卡诺图以后，卡诺图中的0的数目就等于最大项中和项的数目。没有0的小方格对应于表达式的1。通常，对应表达式的1的小方格不填1。如下面的步骤所示，图A.1给出卡诺图填写的过程。

步骤1：确定最大项表达式中每个和项对应的二进制数。就是使得和项为0的二进制数。

步骤2：得到每个和项的二进制数之后，把0放到相应的卡诺图中的小方格里。

例 A.1　把如下的最大项表达式映射到卡诺图中。

$$(\overline{A}+\overline{B}+C+D)(\overline{A}+B+\overline{C}+\overline{D})(A+B+\overline{C}+D)(\overline{A}+\overline{B}+\overline{C}+\overline{D})(A+B+\overline{C}+\overline{D})$$

解：评估如下给出的表达式，把表达式中的和项对应的0放到图A.2的4变量卡诺图中的小方格里。

$$\underset{1100}{(\overline{A}+\overline{B}+C+D)}\underset{1011}{(\overline{A}+B+\overline{C}+\overline{D})}\underset{0010}{(A+B+\overline{C}+D)}\underset{1111}{(\overline{A}+\overline{B}+\overline{C}+\overline{D})}\underset{0011}{(A+B+\overline{C}+\overline{D})}$$

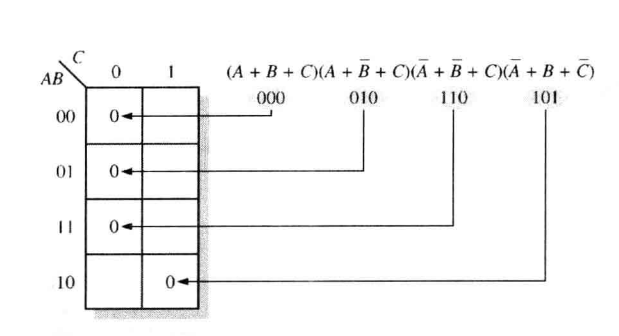


图 A.1　最大项表达式的卡诺图的例子

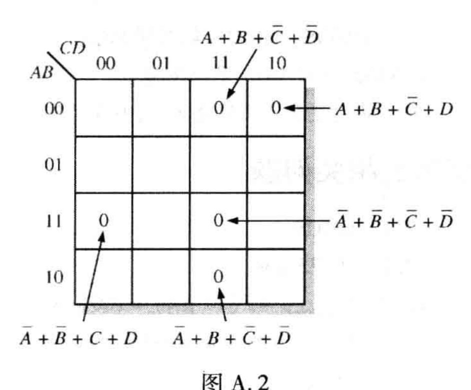


图 A.2

最大项表达式的卡诺图化简

最大项表达式的化简和最小项[标准与或项(SOP)]表达式的化简基本相同，除了圈0的组合产生最简的和项而不是圈1的组合产生最简的乘积项之外。圈0的结果和4.9节介绍的圈1的结果相同。

例 A.2　使用卡诺图化简如下的最大项表达式。

$$(A+B+C)(A+B+\overline{C})(A+\overline{B}+C)(A+\overline{B}+\overline{C})(\overline{A}+\overline{B}+C)$$

同时，推导等效的最小项表达式。

解：表达式对应的二进制数如下：

$$(0+0+0)(0+0+1)(0+1+0)(0+1+1)(1+1+0)$$

把最大项表达式映射到卡诺图上，如图 A.3 所示，对 0 的小方格组合圈组。

注意 110 小方格中的 0 包含在一个有 2 个小方格的组合中，而这个组合里使用了 4 个小方格组合中的 0。每个组合得到的和项如图所示，得到的最简的或与项表达式如下：

$$A(\overline{B}+C)$$

记住，最简的或与项表达式和原来的最大项表达式等价。

为 1 的区域的圈组产生一个与或项表达式，它和圈 0 等价。

$$AC+A\overline{B}=A(\overline{B}+C)$$

例 A.3　使用卡诺图化简如下的最大项表达式。

$$(B+C+D)(A+B+\overline{C}+D)(\overline{A}+B+C+\overline{D})(A+\overline{B}+C+D)(\overline{A}+\overline{B}+C+D)$$

解：第一项必须扩展为 $(\overline{A}+B+C+D)$ 和 $(A+B+C+D)$ 以得到最大或与项表达式，然后进行卡诺图的映射，如图 A.4 所示，对 0 圈组。如图给出每个组的或项，简化的或与项表达式如下：

$$(C+D)(A+B+D)(\overline{A}+B+C)$$

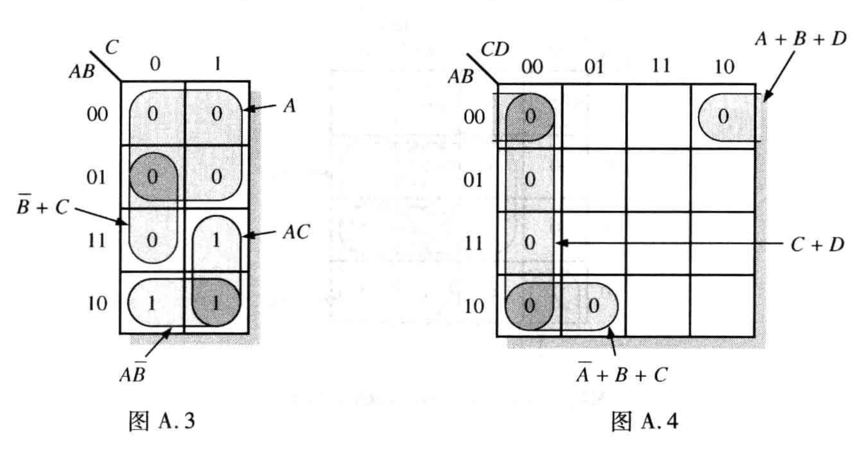


图 A.3　　　图 A.4

使用卡诺图在最大项和最小项之间转换

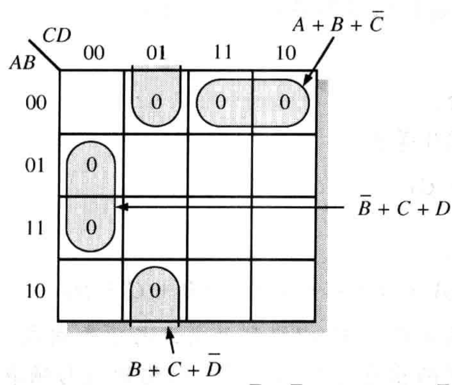
当最大项表达式映射以后，可以很简单地从卡诺图直接把它转换为等价的与或项表达式。同样，对于一个给出最小项表达式，从卡诺图可以直接得到等价的或与项表达式。这就提供了一个很好的比较方法，即确定一个表达式的两种最简形式中，哪一种可以使用最少的门来实现。

对于或与项表达式，所有不包含 0 的小方格都包含 1，从这些 1 可以得到与或项表达式。同样，对于与或项表达式，所有不包含 1 的小方格都包含 0，从这些 0 可以得到或与项表达式。例 A.4 解释了这种转换。

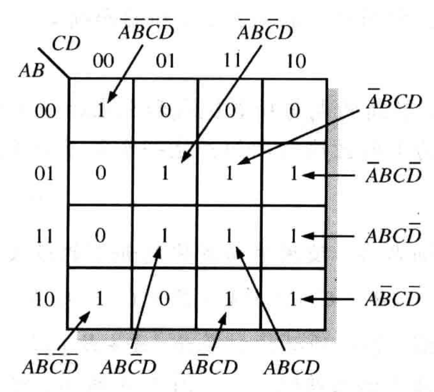
例 A.4　使用卡诺图，把下面的最大项表达式转换为最简或与项表达式、最小项表达式和最简与或项表达式。

$$(\overline{A}+\overline{B}+C+D)(A+\overline{B}+C+D)(A+B+C+\overline{D})$$
$$(A+B+\overline{C}+\overline{D})(\overline{A}+B+C+\overline{D})(A+B+\overline{C}+D)$$

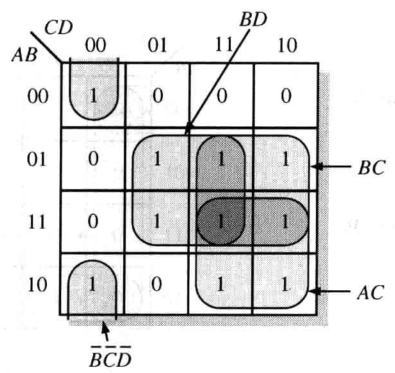
解: 进行最大项表达式对应的 0 映射，并且对 0 圈组，得到如图 A.5(a)所示的最简或与项表达式。在图 A.5(b)中，不包含 0 的小方格中添加 1。从包含 1 的小方格中，得到如图所示的一个最小与项(标准与项，或称包含所有变量的与项)。从这些与项得到最小项表达式。在图 A.5(c)中，对 1 圈组得到最简的与或项表达式。



(a) 最简或与项: $(A+B+C)(\overline{B}+\overline{C}+D)(B+C+\overline{D})$



(b) 标准与或项:
$\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}+\overline{A}B\overline{C}D+\overline{A}BCD+\overline{A}BC\overline{D}+ABC\overline{D}+A\overline{B}C\overline{D}+A\overline{B}\,\overline{C}\,\overline{D}+AB\overline{C}D+A\overline{B}CD+ABCD$



(c) 最简与或项: $AC+BC+BD+\overline{B}\,\overline{C}\,\overline{D}$

图 A.5

附录 B　Q-M 方法(奎恩-麦克拉斯基化简法)

对于少于等于 4 个变量的布尔函数，卡诺图的化简方法非常有效。当有 5 个变量时，使用卡诺图的方法就很困难，对于 5 个以上的变量，使用卡诺图就完全不切实际了。Q-M 方法(奎恩-麦克拉斯基化简法)是对各个项应用布尔分配法找出最简与或项的一种正规的表格方法，其中最简与或项通过消除两项中出现的互补项得到(例如，$ABCD + ABC\overline{D} = ABC$)。与卡诺图方法不同，Q-M 方法比较适合对布尔表达式应用计算机的方法化简，这是它的主要用途。对于一个简单的表达式，例如 4 个变量以内，或许甚至是 5 个变量，卡诺图比较简单。

为了应用 Q-M 方法，首先以最小与或项的形式写出函数式。为了解释起见，使用如下表达式：

$$X = \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}CD + \overline{A}B\overline{C}\,\overline{D} + \overline{A}B\overline{C}D + A\overline{B}C\overline{D} + AB\overline{C}\,\overline{D} + AB\overline{C}D + ABCD$$

然后把函数中各项对应的二进制数放到表 B.1 的真值表中。在函数中的最小项列在右边一列中。

应用 Q-M 方法的第二步是按照最小项中 1 的数目进行分组，安排原表达式中的最小项，如表 B.2 所示。在这个例子中，有 4 个最小项组。(注意，如果 m_0 在表达式中，那么就有 5 个组。)

表 B.1

ABCD	*X*	最小项
0000	0	
0001	1	m_1
0010	0	
0011	1	m_3
0100	1	m_4
0101	1	m_5
0110	0	
0111	0	
1000	0	
1001	0	
1010	1	m_{10}
1011	0	
1100	1	m_{12}
1101	1	m_{13}
1110	0	
1111	1	m_{15}

表 B.2

1 的数目	最小项	*ABCD*
1	m_1	0001
	m_4	0100
2	m_3	0011
	m_5	0101
	m_{10}	1010
	m_{12}	1100
3	m_{13}	1101
4	m_{15}	1111

第三步，比较相邻的组，观察是否有最小项之间只有一个位置的位不同的项。如果有，对于这两个最小项写上复选标记，如表 B.3 所示。需要检查每个最小项和其相邻组中的所有项，不相邻的组中的项不需要检查。在第一级的列中，有最小项名称的列表，以及对应的二进制数，其中 x 所处的位置不同。例如，第一组的最小项 m_1 (0001)和第二组的最小项 m_3 (0011)只有位置 *C* 不同，所以为这两项写上复选标记，并且在第一级列写上 00x1。最小项 m_4 (0100)和最小

项 m_5 (0101)只有 D 位置不同，所以为这两项写上复选标记，并且在第一级列写上 010x。如果一个项对应于两项，那么就写上两次的结果。在此例中，注意 m_1 可以和第二组的 m_5 再应用一次，这次是位置 B 不同，位置 B 写上 x。

表 B.3

最小项中 1 的数目	最小项	$ABCD$	第一级
1	m_1	0001 ✓	(m_1, m_3) 00x1
	m_4	0100 ✓	(m_1, m_5) 0x01
2	m_3	0011 ✓	(m_4, m_5) 010x
	m_5	0101 ✓	(m_4, m_{12}) x100
	m_{10}	1010	(m_5, m_{13}) x101
	m_{12}	1100 ✓	(m_{12}, m_{13}) 110x
3	m_{13}	1101 ✓	(m_{13}, m_{15}) 11x1
4	m_{15}	1111 ✓	

在表 B.3 中，最小项 m_4 和最小项 m_{12} 只有位置 A 不同。在表 B.4 的第一级的列中写上这两个项和 x100。对组 2 和组 3 也完成这样的步骤。在这些组中，m_5 和 m_{13} 与 m_{12} 和 m_{13} 结合(注意 m_{12} 和 m_4 前面已经使用过，这里再一次使用)。对于组 3 和组 4，m_{13} 与 m_{15} 和在一起写在表 B.4 第一级的列中。

在此例中，最小项 m_{10} 没有复选标记，因为没有其他的最小项满足要求。这个最小项称为基本主蕴含项，它必须包含在最终简化的表达式中。

在第一级列出的项已经用于形成一个简化的表(见表 B.4)，表中的组比前面的少一个。记下第一级中剩下的 1 的数目，用来形成 3 个新组。

新组中的项与下面相邻组的项进行比较。仅比较那些相邻组中 x 在相同位置上的项。否则就继续比较。如果两项只在一个位置上的 x 不同，那么和前面一样，对应的两项写上复选标记，所有这些最小项列在第二级的列中。和前面的一样，在第二级中，只有一个位置变化，写上 x。

表 B.4

第一级	第一级中 1 的数目	第二级
(m_1, m_3) 00x1	1	$(m_4, m_5, m_{12}, m_{13})$ x10x
(m_1, m_5) 0x01		$(m_4, m_5, m_{12}, m_{13})$ x10x
(m_4, m_5) 010x ✓		
(m_4, m_{12}) x100 ✓		
(m_5, m_{13}) x101 ✓	2	
(m_{12}, m_{13}) 110x ✓		
(m_{13}, m_{15}) 11x1	3	

对于这里的例子，注意组 1 中的第三项和组 2 中的第二项满足要求，仅在位置 A 不同。组 1 中的第四项和组 2 中的第一项也可以组合，它们形成冗长的最小项集合。其中一个可以在列中删除，因此不会在最后的表达式中用到。

对于复杂的表达式，描述的过程可以继续进行。对于此例，可以从第二级的表达式读出为 $B\overline{C}$。没有标记的项将出现在最后的简化的表达式中。第一个没有标记的项读出为 $\overline{A}\,\overline{B}D$。下一个读出为 $\overline{A}\,\overline{C}D$。最后一个读出为 ABD。回顾一下，m_{10} ($A\overline{B}C\overline{D}$)是基本主蕴含项，所以在最后的表达式中出现。使用没有标记的项得到的简化的表达式是

$$X = B\overline{C} + \overline{A}\,\overline{B}D + \overline{A}\,\overline{C}D + ABD + A\overline{B}C\overline{D}$$

尽管这是一个正确的表达式，但是可能不是最简的表达式。最后检查一下是否还有不必要的项可以去除。表达式中的项写在一个主蕴含表中，如表 B.5 所示，得到每个主蕴含的标注过的最小项。

表 B.5

最佳蕴含项	最小项							
	m_1	m_3	m_4	m_5	m_{10}	m_{12}	m_{13}	m_{15}
$B\overline{C}$ $(m_4, m_5, m_{12}, m_{13})$			✓	✓		✓	✓	
$\overline{A}\,\overline{B}D$ (m_1, m_3)	✓	✓						
$\overline{A}\,\overline{C}D$ (m_1, m_5)	✓			✓				
ABD (m_{13}, m_{15})							✓	✓
$A\overline{B}C\overline{D}$ (m_{10})					✓			

如果一个最小项列中仅有一个复选标记，那么对应的这个主蕴含项就是基本主蕴含项，必须包含在最终的表达式中。项 ABD 必须包含在最终的表达式中，因为 m_{15} 在最小项列中仅有一个复选标记。同样，主蕴含项 $A\overline{B}C\overline{D}$ 对应的 m_{10} 也仅有一个复选标记，所以它也包含在最终的表达式中。注意，主蕴含项 $\overline{A}\,\overline{C}D$ 对应的两个复选标记的最小项在列中的第 1 行和第 2 行上都有复选标记，所以这一项并不是必要的。因此，最终简化的表达式是

$$X = B\overline{C} + \overline{A}\,\overline{B}D + ABD + A\overline{B}C\overline{D}$$

附录 C　数字电路 NI Multisim 仿真
——仿真、样机、测试电路理论、设计与画图①

理论、设计与画图

随着电子电路和系统越来越先进，在电路设计工程中，设计者将依赖于计算机。对于设计者不可缺少的是系统设计、仿真、样机和画出电路图。学生可以利用工程和设计过程来巩固课堂上的概念和理论。学生设计过程的三个阶段包括理解电路理论、设计和电路仿真及通过样机来验证结果(见图 C.1)。

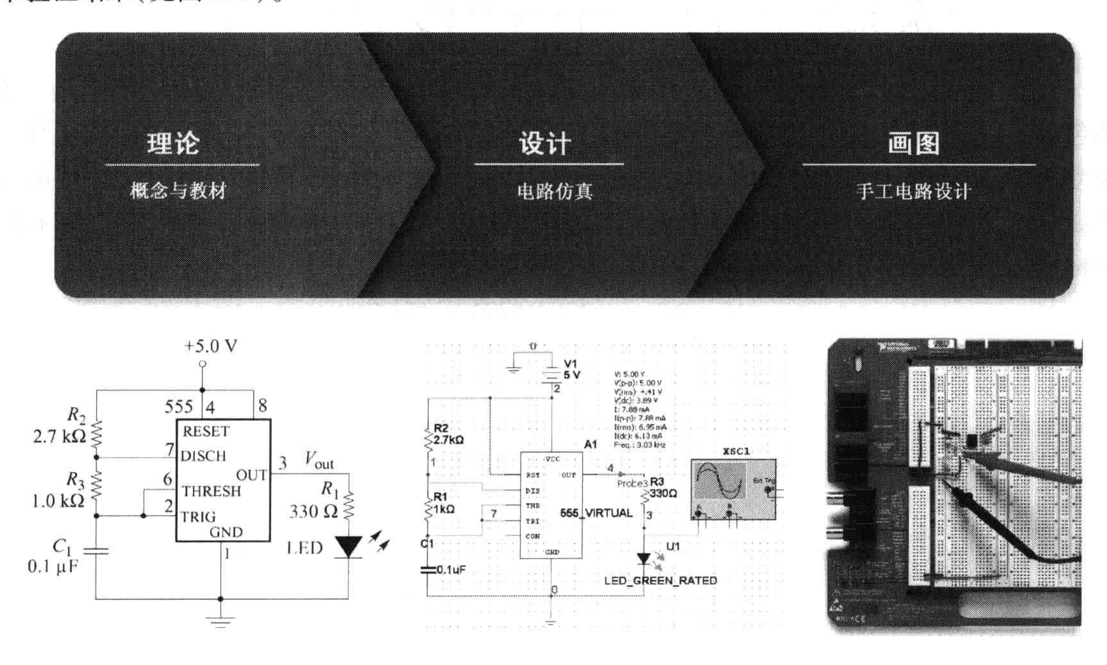


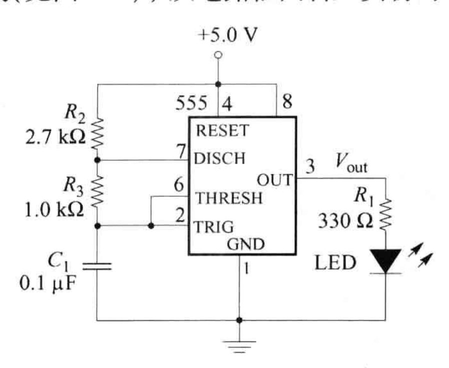
图 C.1　学生设计过程的三个阶段

美国国家仪器公司电子教育平台(National Instruments Electronics Education)是一个点对点的工具链，用于满足学生和教师的需要。该平台包括 NI Multisim 仿真软件、NI 教育实验虚拟套件(NI ELVIS)样机工作站和 NI LabVIEW 图形编程环境。NI Multisim 提供直观示意图捕获和 SPICE 仿真，帮助学生探索电路理论和设计电路并且观察结果。NI ELVIS 是一个样机形式的平台，让学生很容易地创建他们的电路。使用 NI LabVIEW，学生可以测量实时信号和比较仿真结果。

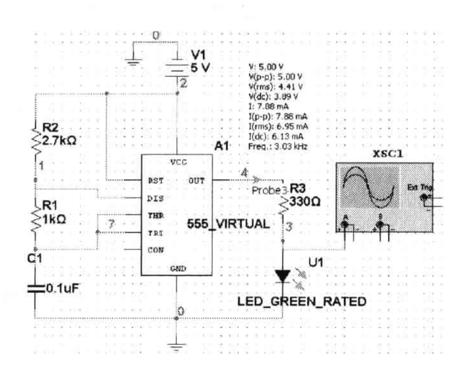
1. 调研。通过数字电子技术教材可以学到电路设计的基础理论。通过下载用于教材的 Multisim 电路文件，然后在使用方便的 Multisim 环境里应用来增强重要概念的理解。使用 Multisim 电路文件建立一个电路性能的深刻理解。使用预先建立的电路文件，仿真和分析本章例题和习题的电路性能。

---

2. 设计和仿真(见图C.2和图C.3)。电路仿真提供一个人机交互的电路界面。起初建立一个电路，使用内置电路仪器和探针学习在理想环境下有关设计的内容。使用Multisim 3D(三维)面包板(见图C.4)，从电路框图转入实物的制作。



图C.2　设计振荡器的示意图



图C.3　Multisim仿真电路

3. 样机、测量和比较。建立实物电路的实际动手经验是必不可少的。由Multisim的3D面包板过渡到NI ELVIS的实物面包板(见图C.4)，学生可以通过样机电路无缝地完成整个设计过程。在LabVIEW环境里，比较实物测量值和仿真值以巩固理论知识，充分理解电路的性能，以及建立专业过程分析的基础。

NI Multisim

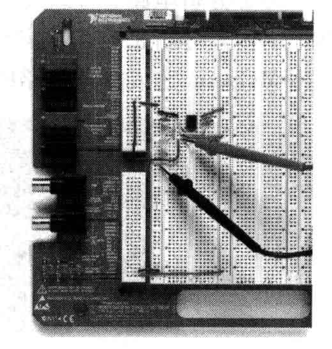
NI Multisim软件集成了功能强大的SPICE仿真和示意图绘制功能，成为一个直观的计算机中的电子实验室。使用Multisim学生版软件，通过如下的特性增加学习内容：

建立在简单和易学环境下的电路

作为作业和预习，仿真和分析电路

在去实验室以前，在家完成3D面包板

使用大约4000个元件的库，完成50个元件以内的电路设计



图C.4　ELVIS样机

在使用工业标准SPICE的背景下，Multisim的拖放界面使得电路的绘制、连线和分析变得简单和易操作。学生可以从零开始建立电路，并且在一个理想的实验环境下，使用内置的虚拟仪器和探针学习有关设计的工作。使用3D面包板，学生可以跳过电路框图进行实物的制作。

使用本教材的NI Multisim　根据教材创建一个例子，以熟悉Multisim环境。首先，运行Multisim，打开一个新的绘图窗口(File ≫ New ≫ Schematic Capture)。通过在电路窗口放置来自元件工具栏的元件，开始设计电路。点击元件工具栏将打开元件浏览器。选择元件的系列，再选择单个元件，然后双击它，放到电路窗口中。

一旦选择了一个元件，它将吸附在鼠标上，并随鼠标移动。在需要绘制元件的位置再次点击鼠标，将放置此元件。Multisim初学者应该使用元件的BASIC _VIRTUAL系列，可以为元件设定任意值。作为一个简单的Multisim电路的例子，图C.5给出了一个时钟振荡器。这个电路连接一个555虚拟定时器，可以在Mix-Virtual系列的Mix组里找到。

下一步是在元件和元件之间连线，简单地在元件的一端单击鼠标左键，再在另一端单击鼠

标左键。Multisim 将自动为两个元件之间的虚拟连线选择最佳路径。始终要确保电路绘制完整，接着才可以进行仿真。然后，仿真的结果可以和实物电路比较。

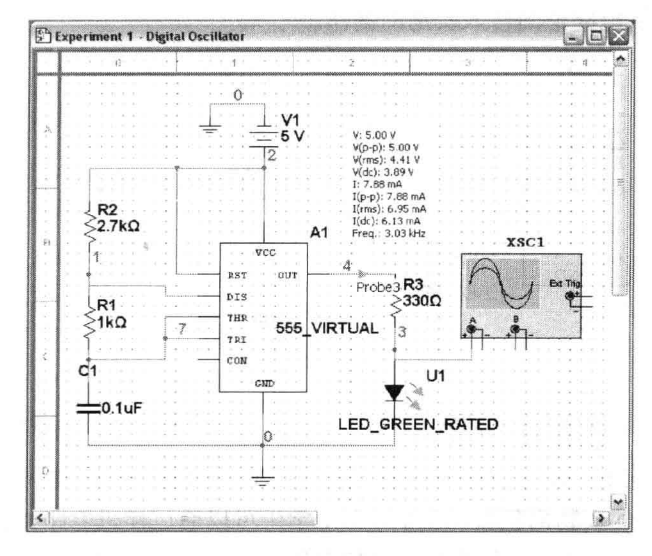


图 C.5 Multisim 振荡器的绘图

为了分析电路，可以运行仿真步骤，使用测量探针测量来自电路的电压和其他特性。使用虚拟示波器分析来自 555 定时器电路的输出信号。使用 3D 面包板(见图 C.6)跳过电路框图，使用 Multisim 完成实物电路的工作。

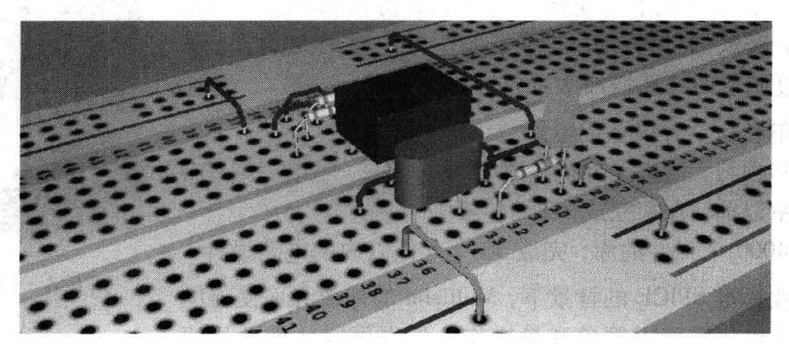


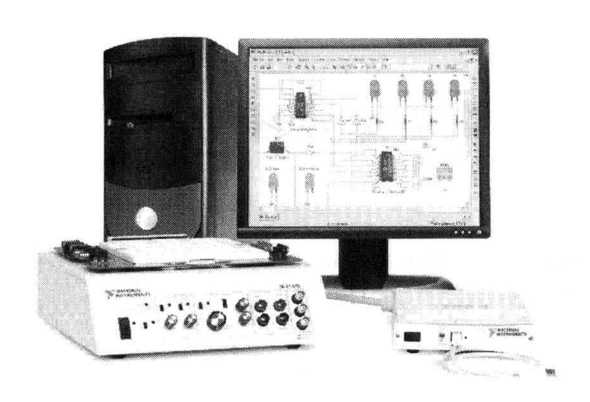
图 C.6 设计振荡器的 NI ELVIS 3D 样机

NI ELVIS

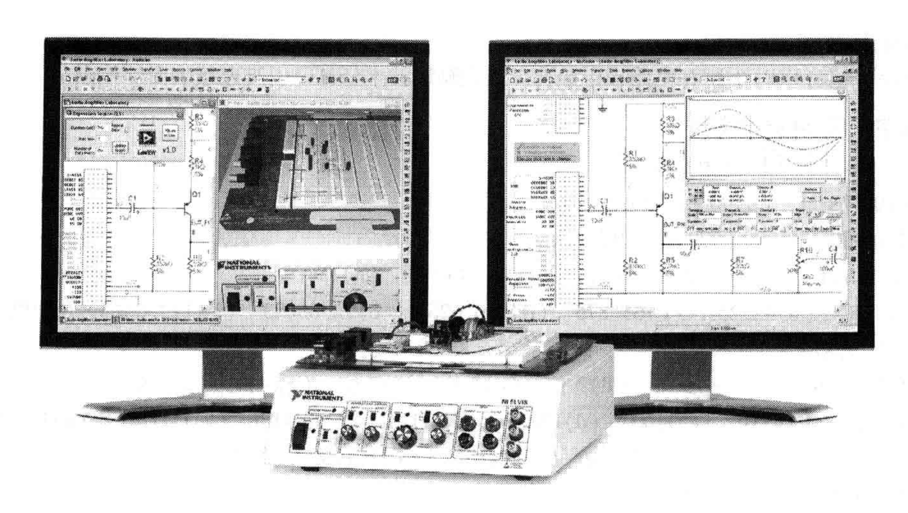
NI ELVIS(见图 C.7 和图 C.8)是一个基于 LabVIEW 设计和样机的环境，用于高校的科学和工程实验室。NI ELVIS 包括基于 LabVIEW 的虚拟仪器、多函数数据采集设备、传统设计的桌面工作平台和样机版。这个组合体提供一个可用于所有教学实验室的仪器套件。

NI ELVIS 是所有实验室的一个理想伙伴，它使用容易建立、维修和携带的 NI ELVIS 特性的 USB 连接，还有一个用于实物电路样机的便携式面包板。

NI ELVIS 提供一个面包板样机环境，具有一个包含函数发生器的内置仪器、数字万用表(DMM)、振荡器和各种虚拟电源。面包板可分离，允许学生在他们的工程和实验室中独立于 NI ELVIS 单元操作。NI ELVIS 提供用于和虚拟仪器接口的、基于 LabVIEW 的软件。这些仪器可以修改用于 Multisim 数据下载，以及用于仿真和测量数据的快速比较。



图C.7 NI ELVIS工作站



图C.8

NI Multisim电路文件 下载Multisim电路文件，深刻理解电路性能。从每章结尾的例子和习题中深刻理解电路性能。下载预先建立的文件，可访问：ni.com/academic/floyd。

NI Multisim资源 为了帮助读者开始使用NI Multisim，提供如下的参考资源的链接：

阅读Multisim初学指导
学习Multisim 3小时学习指南
在一个在线论坛上讨论Multisim
下载免费30天评估版Multisim

奇数编号习题答案

第 1 章　基本概念

1. 数字信号可以更有效和更可靠地传输和存储。
3. 时钟，体温计，速度仪。
5. (a)11010001　(b)000101010
7. (a)550 ns　(b)600 ns　(c)2.7 μs　(d)10 V
9. 250 Hz
11. 50%
13. 8 μs; 1 μs
15. 双引线封装(DIP)引脚在电路板中穿孔。表面贴装技术(SMT)引脚连接表面焊接点。

第 2 章　数字系统、运算和编码

1. (a)1　(b)100　(c)100 000
3. (a)400; 70; 1　(b)9000; 300; 50; 6
   (c)100 000; 20 000; 5000; 0; 0; 0
5. (a)3　(b)4　(c)7　(d)8　(e)9　(f)12　(g)11　(h)15
7. (a)51.75　(b)42.25　(c)65.875　(d)120.625　(e)92.65625　(f)113.0625　(g)90.625
   (h)127.96875
9. (a)5 位　(b)6 位　(c)6 位　(d)7 位　(e)7 位　(f)7 位　(g)8 位　(h)8 位
11. (a)1010　(b)10001　(c)11000　(d)110000　(e)111101　(f)1011101　(g)1111101　(h)10111010
13. (a)1111　(b)10101　(c)11100　(d)100010　(e)101000　(f)111011
    (g)1000001　(h)1001001
15. (a)100　(b)100　(c)1000　(d)1101　(e)1110　(f)11000
17. (a)1001　(b)1000　(c)100011　(d)110110　(e)10101001　(f)10110110
19. 全 0 或全 1。
21. (a)010　(b)001　(c)0101　(d)00101000　(e)0001010　(f)11110
23. (a)00011101　(b)11010101　(c)01100100　(d)11111011
25. (a)00001100　(b)10111100　(c)01100101　(d)10000011
27. (a) −102　(b) +116　(c) −64
29. (a)0 10001101 11110000101011000000000　(b)1 10001010 11000001100000000000000
31. (a)00110000　(b)00011101　(c)11101011　(d)100111110
33. (a)11000101　(b)11000000
35. 100111001010
37. (a)00111000　(b)01011001　(c)101000010100　(d)010111001000　(e)0100000100000000
    (f)1111101100010111　(g)1000101010011101
39. (a)35　(b)146　(c)26　(d)141　(e)243　(f)235　(g)1474　(h)1792
41. (a)60_{16}　(b)$10B_{16}$　(c)$1BA_{16}$
43. (a)10　(b)23　(c)46　(d)52　(e)67　(f)367　(g)115　(h)532　(i)4085
45. (a)001011　(b)101111　(c)001000001　(d)011010001　(e)101100000　(f)100110101011
    (g)001011010111001　(h)100101110000000　(i)0010000000010001011
47. (a)00010000　(b)00010011　(c)00011000　(d)00100001　(e)00100101　(f)00110110

(g)01000100 (h)01010111 (i)01101001 (j)10011000 (k)000100100101 (l)000101010110

49. (a)000100000100 (b)000100101000 (c)000100110010 (d)000101010000 (e)000110000110 (f)001000010000 (g)001101011001 (h)010101000111 (i)0001000001010001

51. (a)80 (b)237 (c)346 (d)421 (e)754 (f)800 (g)978 (h)1683 (i)9018 (j)6667

53. (a)00010100 (b)00010010 (c)00010111 (d)00010110 (e)01010010 (f)000100001001 (g)000110010101 (h)0001001001101001

55. 当格雷码在序列中每次从一个数变到下一个数时，格雷码仅有一位发生变化。

57. B 不正确。

59. (a)110100100 (b)000001001 (c)111111110

61. 在每种情况下，得到其他的数。

63. 余数是 0100，指出一个错误。

第 3 章 逻辑门

1. 参见图 P-1。

3. 参见图 P-2。

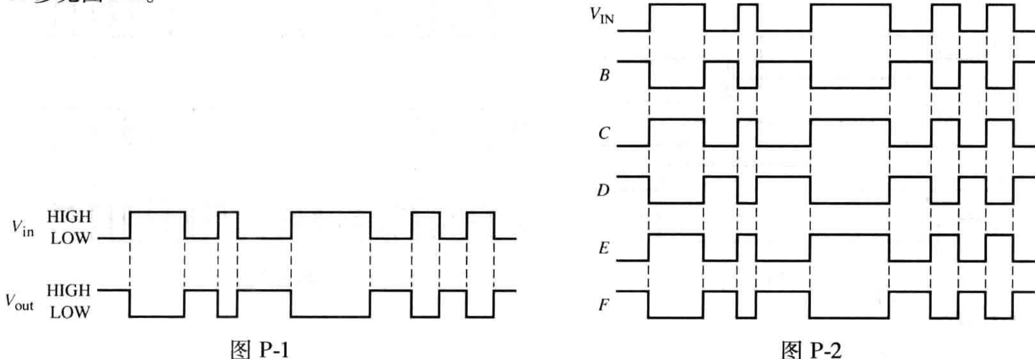


图 P-1 图 P-2

5. 参见图 P-3。

7. 参见图 P-4。

9. 参见图 P-5。

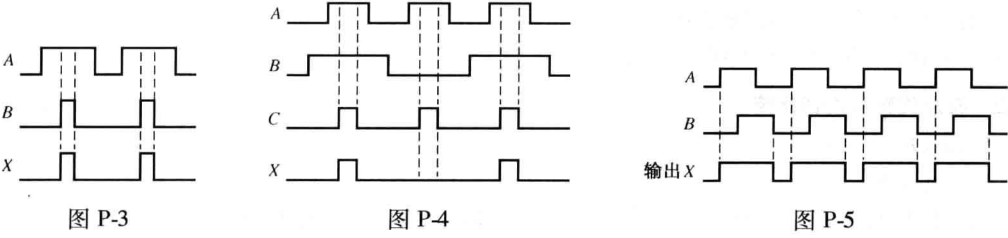


图 P-3 图 P-4 图 P-5

11. 参见图 P-6。

13. 参见图 P-7。

15. 参见图 P-8。

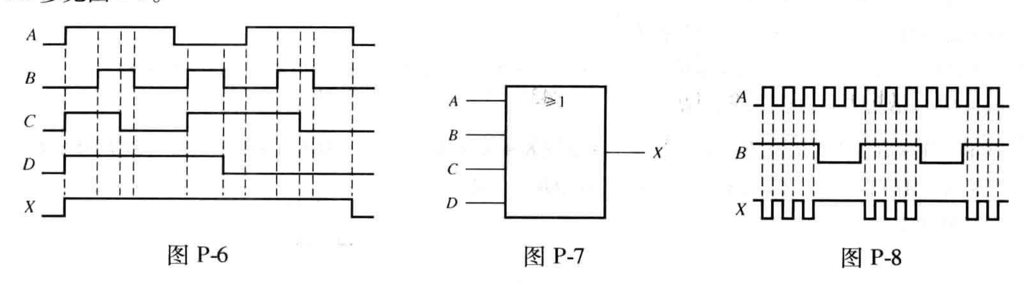


图 P-6 图 P-7 图 P-8

17. 参见图 P-9。

19. 参见图 P-10。

21. 参见图 P-11。

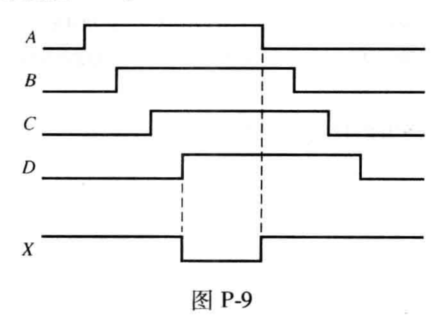


图 P-9

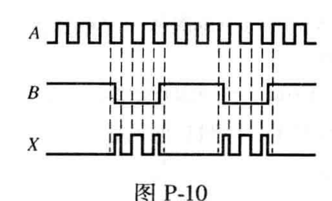


图 P-10

23. XOR $= A\overline{B} + \overline{A}B$; OR $= A + B$

25. 参见图 P-12。

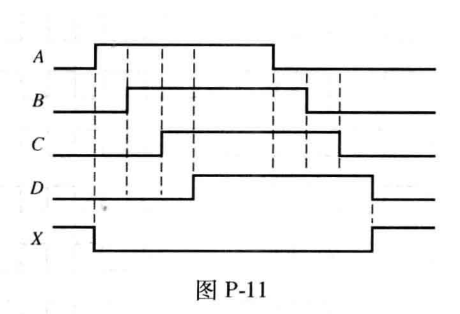


图 P-11

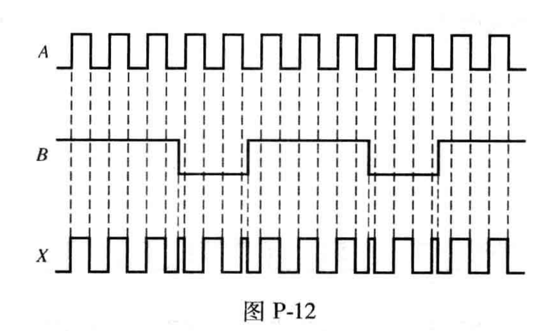


图 P-12

27. CMOS

29. $t_{PLH} = 4.3$ ns; $t_{PHL} = 10.5$ ns

31. 20 mW

33. 图(b)、(c)、(e)中的门是错误的。

35. (a)测试输出(下降 低电平或打开); (b)引脚 4 输入或引脚 6 输出内部开路。

37. 座位安全带输入到与门的连接开路。

39. $X_1 = \overline{A}B$, $X_2 = \overline{A}\,\overline{B}$, $X_3 = A\overline{B}$

第 4 章 布尔代数和逻辑化简

1. $X = A + B + C + D$

3. $X = \overline{A} + \overline{B} + \overline{C}$

5. (a)当 $A = 1$, $B = 1$ 时, $AB = 1$　　(b)当 $A = 1$, $B = 0$, $C = 1$ 时, $A\overline{B}C = 1$
   (c)当 $A = 0$, $B = 0$ 时, $A + B = 0$　　(d)当 $A = 1$, $B = 0$, $C = 1$ 时, $\overline{A} + B + \overline{C} = 0$
   (e)当 $A = 1$, $B = 1$, $C = 0$ 时, $\overline{A} + \overline{B} + C = 0$　　(f)当 $A = 1$, $B = 0$ 时, $\overline{A} + B = 0$
   (g)当 $A = 1$, $B = 0$, $C = 0$ 时, $A\overline{B}\,\overline{C} = 1$

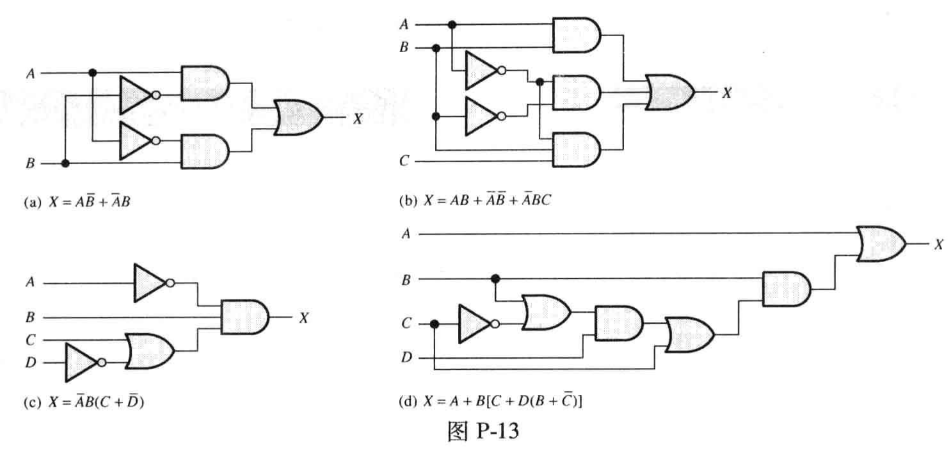
7. (a)交换律　(b)交换律　(c)分配律

9. (a) $\overline{A}B$　(b) $A + \overline{B}$　(c) $\overline{AB}\,\overline{C}$　(d) $\overline{A} + \overline{B} + \overline{C}$　(e) $\overline{A} + \overline{B}\,\overline{C}$　(f) $\overline{A} + \overline{B} + \overline{C} + \overline{D}$
   (g) $(\overline{A} + \overline{B})(\overline{C} + \overline{D})$　(h) $\overline{A}B + C\overline{D}$

11. (a) $(\overline{A} + \overline{B} + \overline{C})(\overline{E} + \overline{F} + \overline{G})(\overline{H} + \overline{I} + \overline{J})(\overline{K} + \overline{L} + \overline{M})$　(b) $\overline{AB}\,\overline{C} + BC$　(c) $\overline{ABCDEFGH}$

13. (a) $X = ABCD$　(b) $X = AB + C$　(c) $X = \overline{\overline{A}B}$　(d) $X = (A + B)C$

15. 参见图 P-13。



(a) $X = A\bar{B} + \bar{A}B$

(b) $X = AB + \bar{A}\bar{B} + \bar{A}BC$

(c) $X = \bar{A}B(C + \bar{D})$

(d) $X = A + B[C + D(B + \bar{C})]$

图 P-13

17. (a) 参见表 P-1。

表 P-1

输入			输出
$\overline{VCR}$	$\overline{CAMI}$	$\overline{RDY}$	$\overline{RECORD}$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

17. (b) 参见表 P-2。

表 P-2

输入			输出
$\overline{RTS}$	$ENABLE$	$\overline{BUSY}$	$\overline{SEND}$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

19. (a) A (b) AB (c) C (d) A (e) $\bar{A}C + \bar{B}C$

21. (a) $BD + BE + DF$ (b) $ABC + ABD$ (c) B (d) $AB + CD$ (e) ABC

23. (a) $A\bar{B} + AC + BC$ (b) $AC + \bar{B}C$ (c) $AB + AC$

25. (a) 变量域：A, B, C 最小项：$A\bar{B}C + A\bar{B}\,\bar{C} + ABC + \bar{A}BC$

(b) 变量域：A, B, C 最小项：$ABC + A\bar{B}C + \bar{A}\,\bar{B}C$

(c) 变量域：A, B, C 最小项：$ABC + AB\bar{C} + A\bar{B}C$

27. (a) 101 + 100 + 111 + 011 (b) 111 + 101 + 001 (c) 111 + 110 + 101

29. (a) $(A + B + C)(A + B + \bar{C})(A + \bar{B} + C)(\bar{A} + \bar{B} + C)$

(b) $(A + B + C)(A + \bar{B} + C)(A + \bar{B} + \bar{C})(\bar{A} + B + C)(\bar{A} + \bar{B} + C)$

(c) $(A + B + C)(A + B + \bar{C})(A + \bar{B} + C)(A + \bar{B} + \bar{C})(\bar{A} + B + C)$

31. (a) 参见表 P-3。

表 P-3

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

31. (b) 参见表 P-4。

表 P-4

X	Y	Z	Q
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

33. (a)参见表 P-5。

表 P-5

A	B	C	X
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

33. (b)参见表 P-6。

表 P-6

W	X	Y	Z	Q
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

35. (a)参见表 P-7。

表 P-7

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

35. (b)参见表 P-8。

表 P-8

A	B	C	D	X
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

37. 参见图 P-14。

39. 参见图 P-15。

41. (a)没有化简 (b)AC (c) $\overline{D}\overline{F} + E\overline{F}$

43. (a)$AB + AC$ (b)$A + BC$ (c) $B\overline{C}D + A\overline{C}D + BC\overline{D} + AC\overline{D}$ (d) $A\overline{B} + CD$

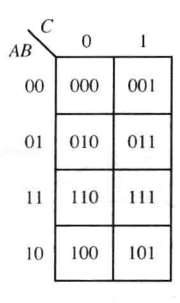


图 P-14

AB \ C	0	1
00	$\overline{A}\overline{B}\overline{C}$	$\overline{A}\overline{B}C$
01	$\overline{A}B\overline{C}$	$\overline{A}BC$
11	$AB\overline{C}$	ABC
10	$A\overline{B}\overline{C}$	$A\overline{B}C$

图 P-15

45. $\overline{B} + C$

47. $\overline{A}\,\overline{B}\,\overline{C}D + C\overline{D} + BC + A\overline{D}$

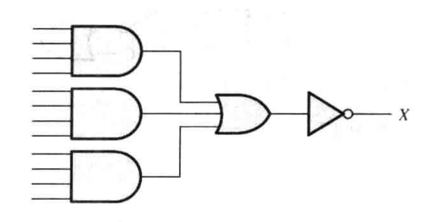
49. $X = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\,\overline{E} + \overline{A}BC\overline{D}E + AB\overline{C}\,\overline{D}\,\overline{E} + \overline{A}\,\overline{B}DE + \overline{A}BD\overline{E} + \overline{B}\,\overline{C}DE + AB\overline{C}D$

51. LED。LED 发射光，LCD 不发射。

53. 少一个反相器和一个门。

第 5 章　组合逻辑分析

1. 参见图 P-16。



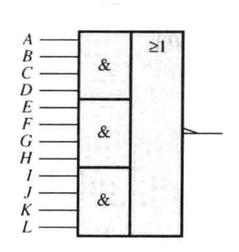


图 P-16

3. (a) $X = ABB$　(b) $X = AB + B$　(c) $X = \overline{A} + B$　(d) $X = (A + B) + AB$　(e) $X = \overline{\overline{A}BC}$
(f) $X = (A + B)(\overline{B} + C)$

5. (a)

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

5. (b)

A	B	X
0	0	0
0	1	1
1	0	0
1	1	1

5. (c)

A	B	X
0	0	1
0	1	1
1	0	0
1	1	1

5. (d)

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

5. (e)

A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

5. (f)

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

7. $X = \overline{A\overline{B} + \overline{A}B} = (\overline{A} + B)(A + \overline{B})$

9. $\overline{ABCD + EFGH}$

11. 参见图 P-17。

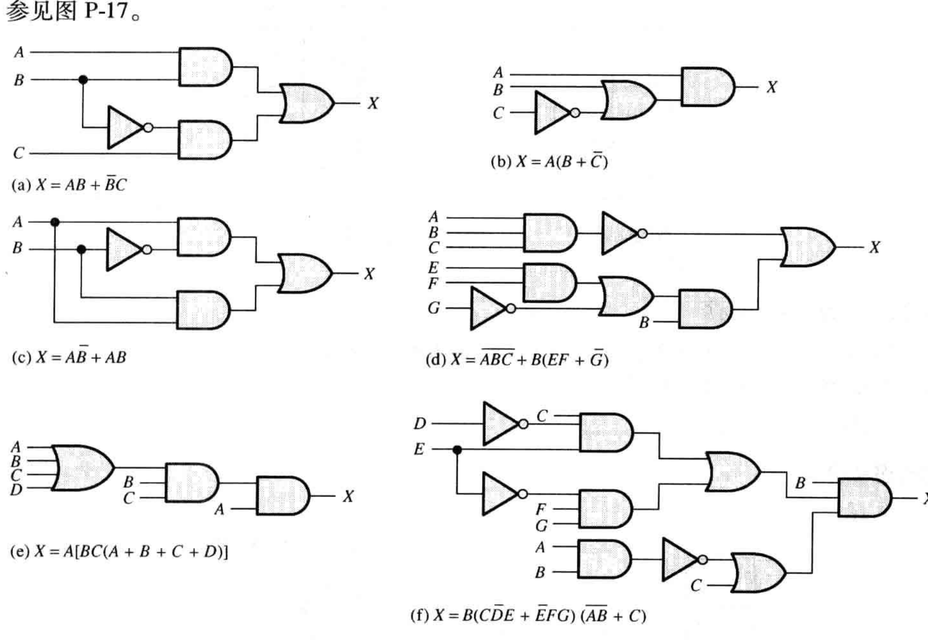


图 P-17

13. 参见图 P-18。

15. $X = AB$

17. (a) 没有化简 (b)没有化简 (c) $X = A$

(d) $X = \overline{A} + \overline{B} + \overline{C} + EF + \overline{G}$ (e) $X = ABC$

(f) $X = BC\overline{D}E + \overline{A}B\overline{E}FG + BC\overline{E}FG$

$X = AB + \overline{C}$

图 P-18

19. (a) $X = AC + AD + BC + BD$

(b) $X = \overline{A}CD + \overline{B}CD$

(c) $X = ABD + CD + E$

(d) $X = \overline{A} + B + D$

(e) $X = ABD + \overline{C}D + \overline{E}$

(f) $X = \overline{AC} + \overline{AD} + \overline{BC} + \overline{BD} + \overline{EG} + \overline{EH} + \overline{FG} + \overline{FH}$

21. 参见图 P-19。

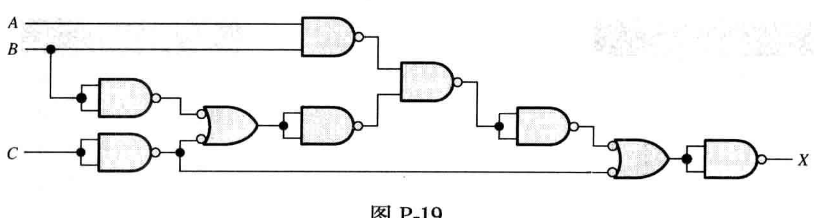


图 P-19

23. 参见图 P-20。

25. 参见图 P-21。

27. 参见图 P-22。

29. $X = A + \overline{B}$; 参见图 P-23。

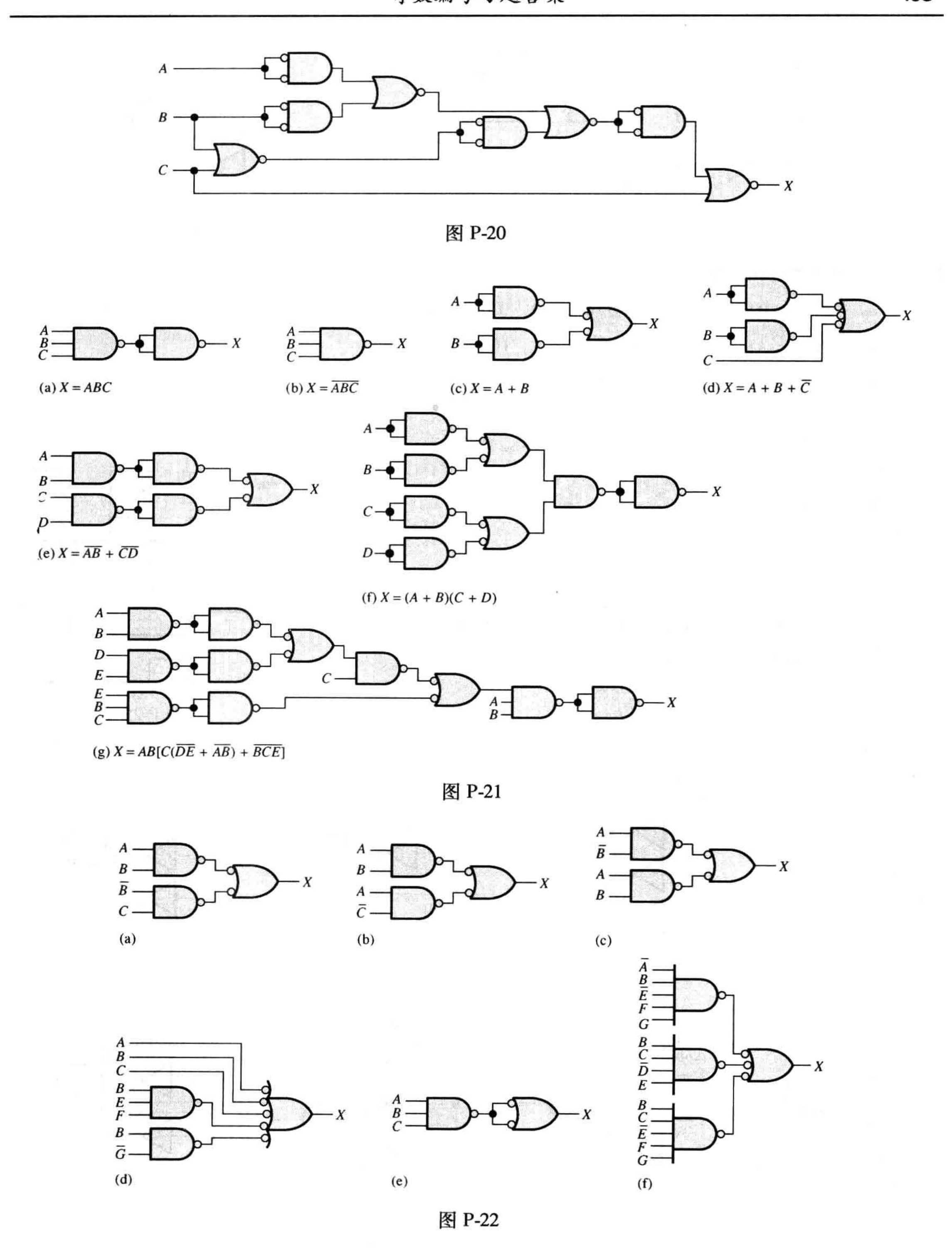


图 P-20

图 P-21

图 P-22

31. $X = \overline{A}\overline{B}\overline{C}$；参见图 P-24。

33. 输出脉冲宽度比规定的最小值大。

35. $X = ABC + D\overline{E}$ 。因为 X 和 G_3 输出相同，G_1 或 G_2 失败，输出下降为低电平。

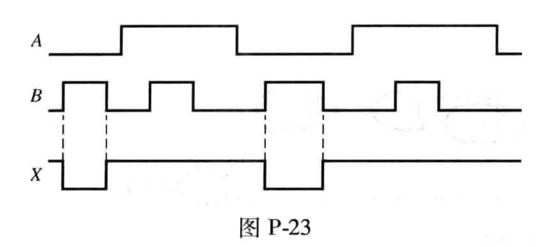


图 P-23

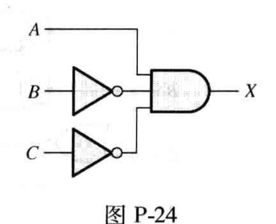


图 P-24

第 6 章 组合逻辑电路函数

1. (a) $A \oplus B = 0$, $\Sigma = 1$, $(A \oplus B)C_{in} = 0$, $AB = 1$, $C_{out} = 1$

   (b) $A \oplus B = 1$, $\Sigma = 0$, $(A \oplus B)C_{in} = 1$, $AB = 0$, $C_{out} = 1$

   (c) $A \oplus B = 1$, $\Sigma = 1$, $(A \oplus B)C_{in} = 0$, $AB = 0$, $C_{out} = 0$

3. (a) $\Sigma = 1$, $C_{out} = 0$;　　(b) $\Sigma = 1$, $C_{out} = 0$;

   (c) $\Sigma = 0$, $C_{out} = 1$;　　(d) $\Sigma = 1$, $C_{out} = 1$

5. 11100

7. $\Sigma_3\Sigma_2\Sigma_1\Sigma_0 = 1101$

9. $\Sigma_1 = 0110$; $\Sigma_2 = 1011$; $\Sigma_3 = 0110$;

   $\Sigma_4 = 0001$; $\Sigma_5 = 1000$

11. 225 ns

13. 当 $A_0 = B_0$ 和 $A_1 = B_1$ 时, $A = B$ 是高电平。参见图 P-25。

15. (a) $A > B = 1$; $A = B = 0$; $A < B = 0$

    (b) $A < B = 1$; $A = B = 0$; $A > B = 0$

    (c) $A = B = 1$; $A < B = 0$; $A > B = 0$

17. 参见图 P-26。

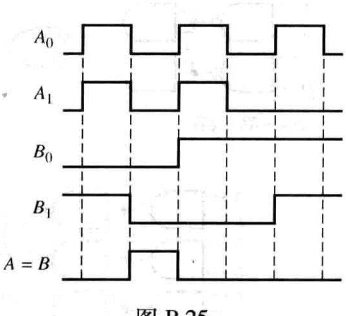


图 P-25

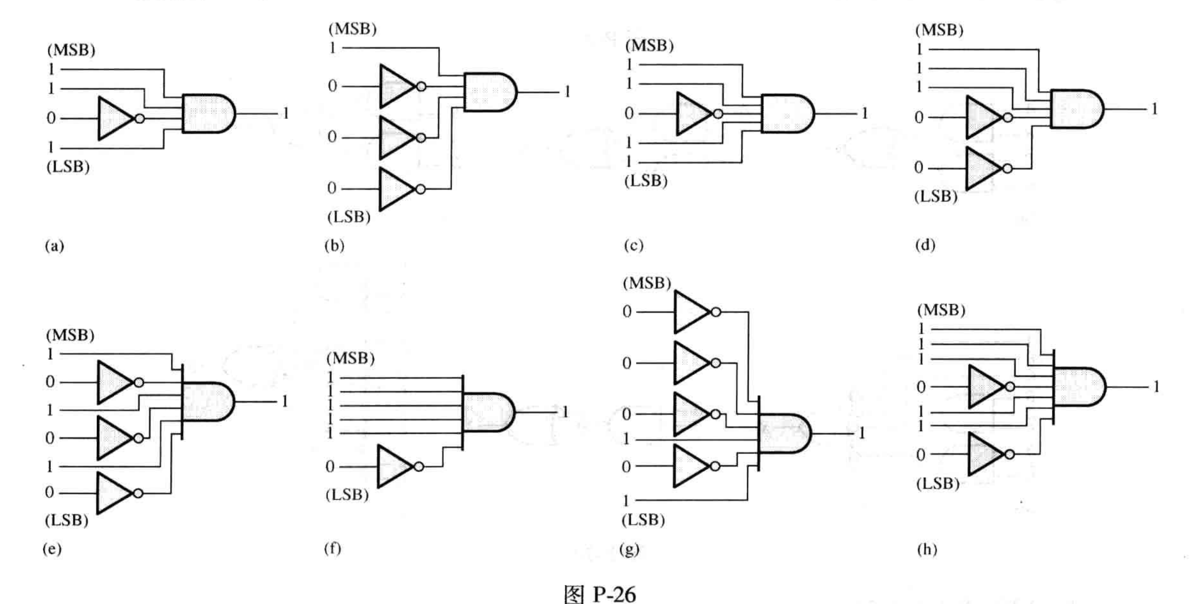


图 P-26

19. $X = A_3A_2\overline{A}_1\overline{A}_0 + \overline{A}_3\overline{A}_2\overline{A}_1A_0 + A_3\overline{A}_2A_1$

21. 参见图 P-27。

23. $A_3A_2A_1A_0 = 1011$, 无效的 BCD 码。

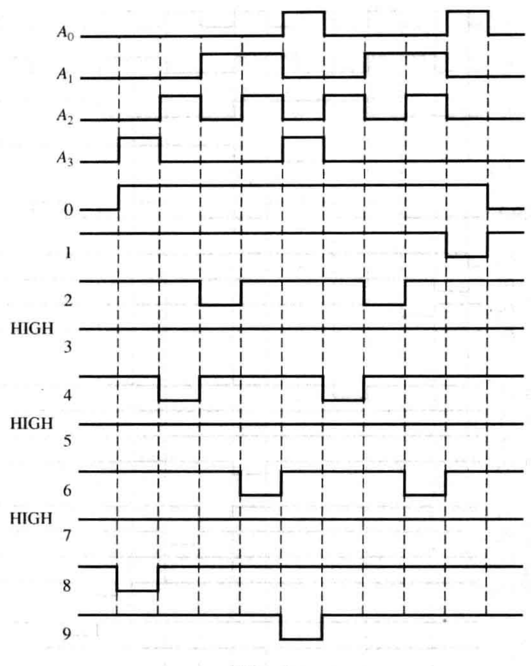


图 P-27

25. (a) $2 = 0010 = 0010_2$　(b) $8 = 1000 = 1000_2$　(c) $13 = 00010011 = 1101_2$
(d) $26 = 00100110 = 11010_2$　(e) $33 = 00110011 = 100001_2$

27. (a) 1010000000 格雷码 → 1100000000 二进制数　(b)0011001100 格雷码→ 0010001000 二进制数
(c)1111000111 格雷码 → 1010000101 二进制数　(d)0000000001 格雷码 → 0000000001 二进制数
参见图 P-28。

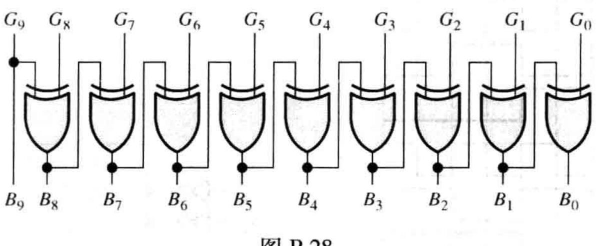


图 P-28

29. 参见图 P-29。

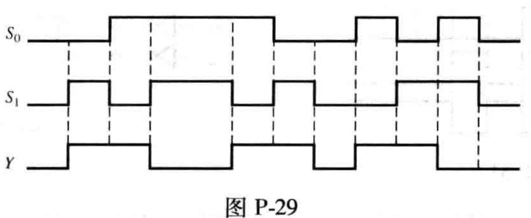


图 P-29

31. 参见图 P-30。

33. 参见图 P-31。

35. 参见图 P-32。

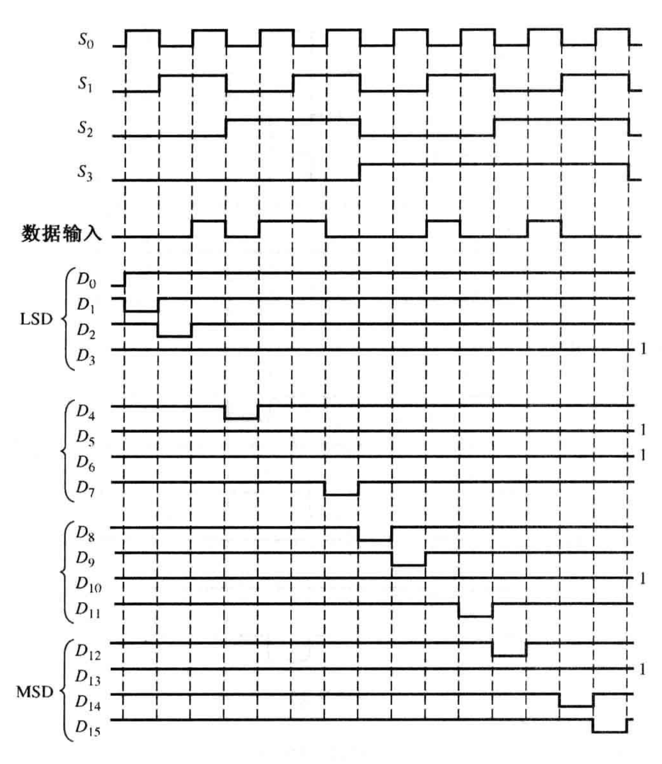


图 P-30

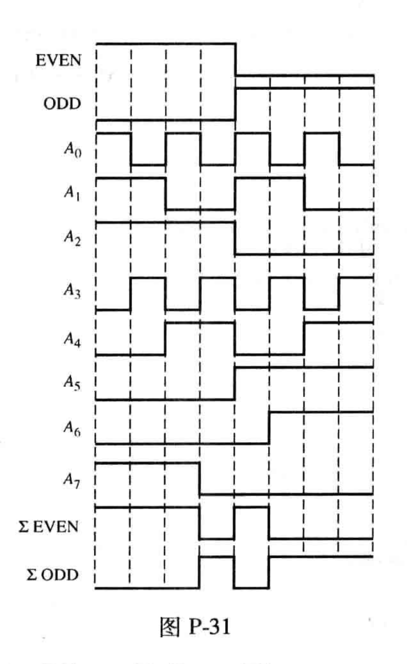


图 P-31

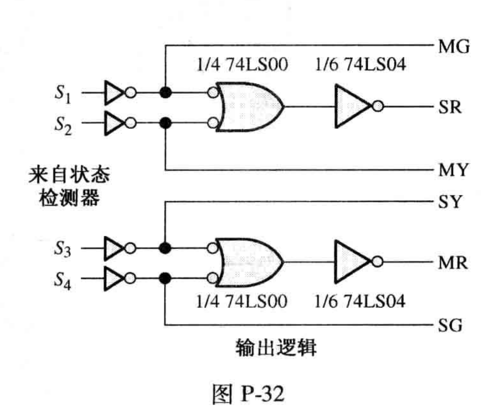


图 P-32

37. $\Sigma = \bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}_{in} + A\bar{B}\bar{C}_{in} + ABC_{in}$ $C_{out} = \bar{A}BC_{in} + A\bar{B}C_{in} + AB\bar{C}_{in} + ABC_{in}$。参见图 P-33。

39. 参见图 P-34 的框图。

41. 参见图 P-35。

43. 参见图 P-36。

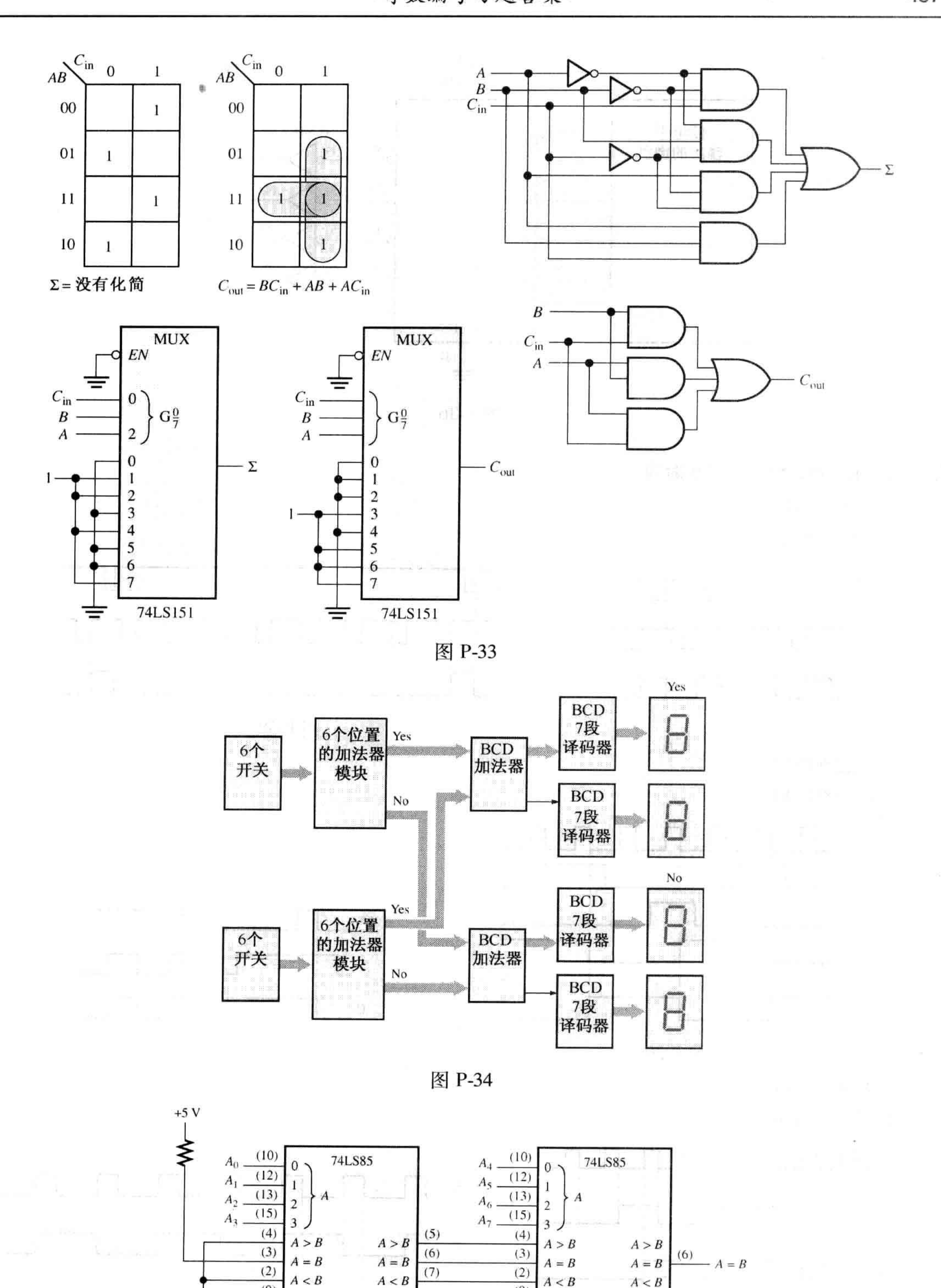


图 P-33

图 P-34

图 P-35

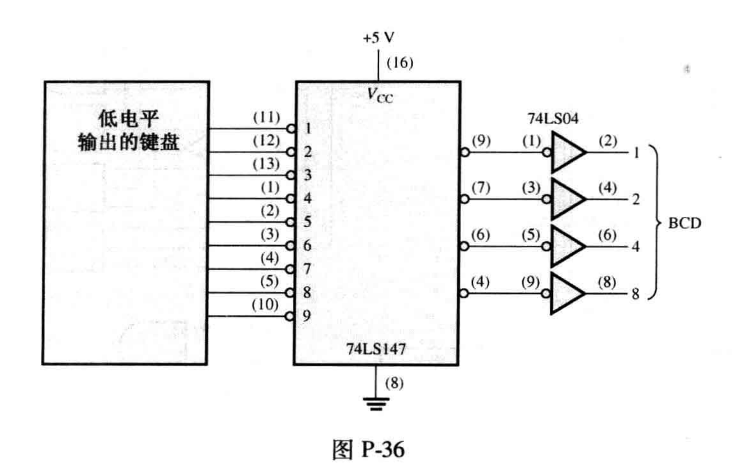


图 P-36

第 7 章 锁存器、触发器和定时器

1. 参见图 P-37。
3. 参见图 P-38。

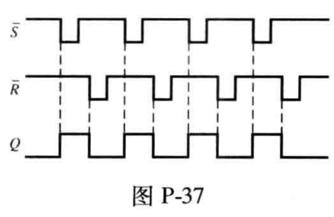


图 P-37

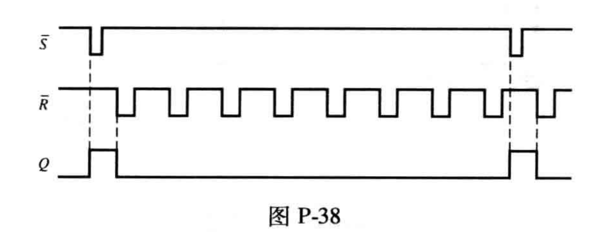


图 P-38

5. 参见图 P-39。
7. 参见图 P-40。

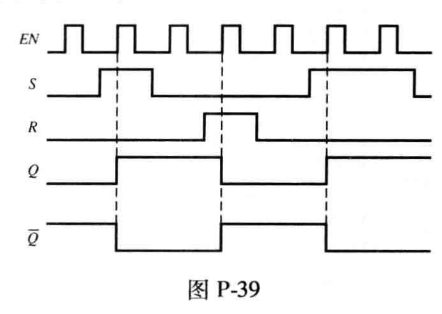


图 P-39

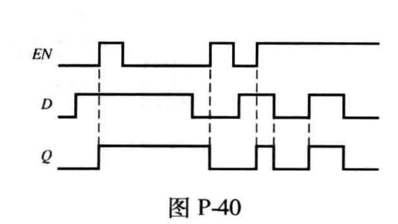


图 P-40

9. 参见图 P-41。
11. 参见图 P-42。

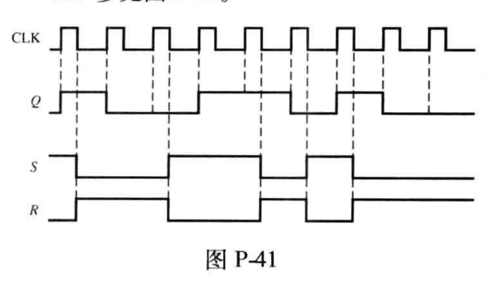


图 P-41

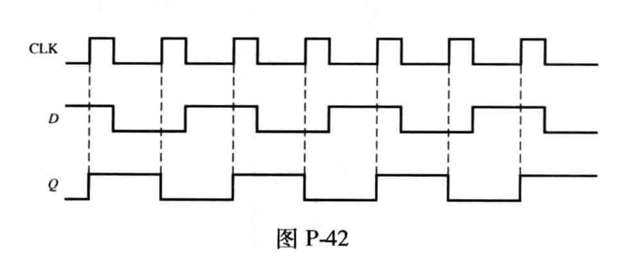


图 P-42

13. 参见图 P-43。

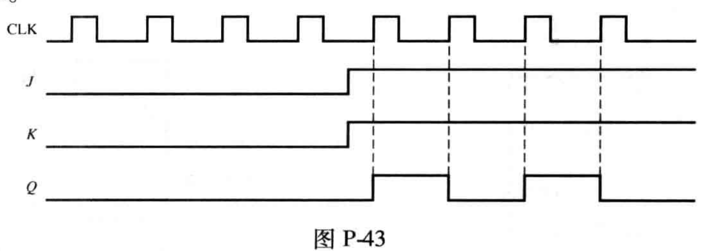


图 P-43

15. 参见图 P-44。

17. 参见图 P-45。

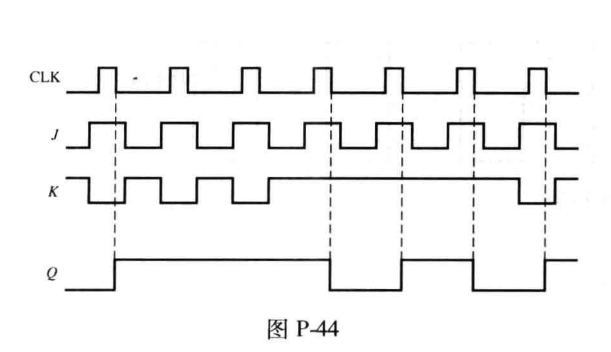


图 P-44

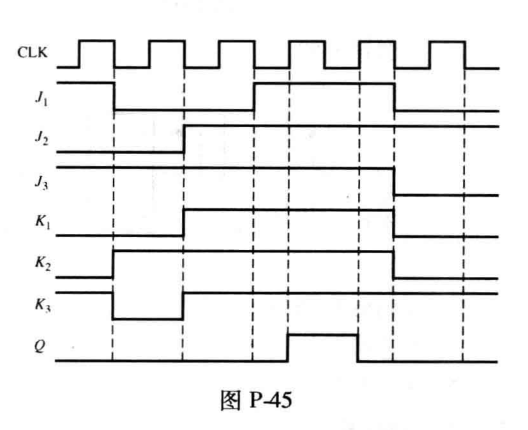


图 P-45

19. 指出电流和直流电源电压的方向。

21. 14.9 MHz

23. 150 mA, 750 mW

25. 除以 2; 参见图 P-46。

27. 4.62 μs

29. C_1 = 1 μF, R_1 = 227 kΩ(使用 220 kΩ)。参见图 P-47。

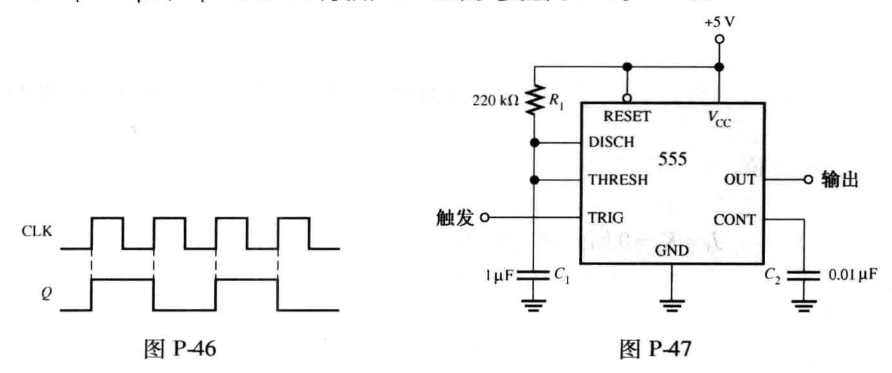


图 P-46 图 P-47

31. 参见图 P-48。

6 s: C_1 = 1 μF, R_1 = 5.5 MΩ (使用 5.6 MΩ)

40 s: C_1 = 2.2 μF, R_1 = 16.5 MΩ (使用 15 MΩ)

33. 参见图 P-49。

6 s: C_{EXT} = 1 μF, R_{EXT} = 18 MΩ

40 s: C_{EXT} = 10 μF, R_{EXT} = 12 MΩ

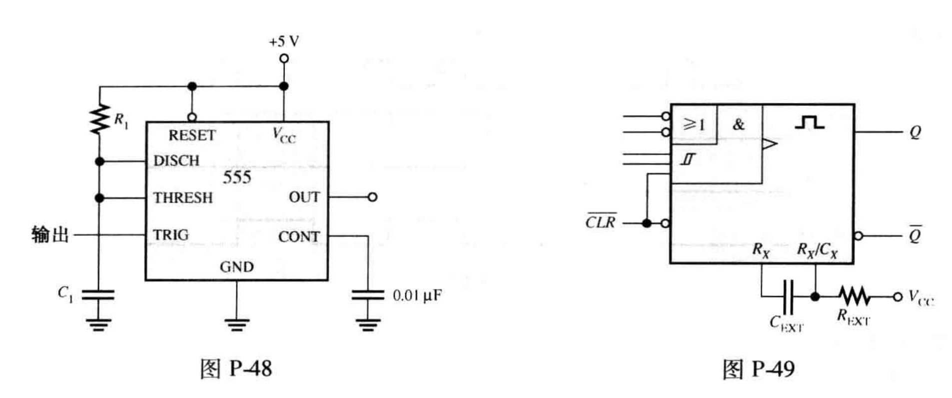


图 P-48　　　　图 P-49

35. 参见图 P-50。

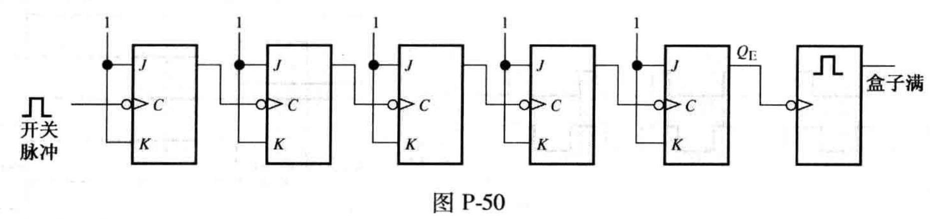


图 P-50

37. Q 输出 U_1 开路。

39. $\overline{SET}$ 输入 U_1 开路。

41. K 输入 U_2 开路。

第 8 章　计数器

1. 参见图 P-51。

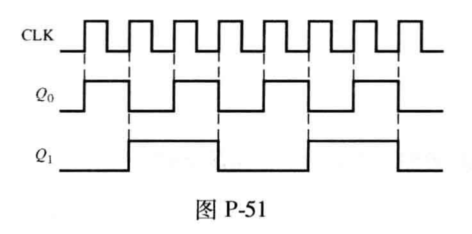


图 P-51

3. 最差情况时的延迟是 24 ns；这种情况发生在当所有触发器的状态从 011 变到 100 或从 111 变到 000。

5. 8 ns

7. 初始时，每个触发器都复位。

在时钟 CLK1 的作用下：

$J_0 = K_0 = 1$ 因此 Q_0变到 1。　　$J_1 = K_1 = 0$ 因此 Q_1保持 0。

$J_2 = K_2 = 0$ 因此 Q_2保持 0。　　$J_3 = K_3 = 0$ 因此 Q_3保持 0。

在时钟 CLK2 的作用下：

$J_0 = K_0 = 1$ 因此 Q_0变到 0。　　$J_1 = K_1 = 1$ 因此 Q_1变到 1。

$J_2 = K_2 = 0$ 因此 Q_2保持 0。　　$J_3 = K_3 = 0$ 因此 Q_3保持 0。

在时钟 CLK3 的作用下：

$J_0 = K_0 = 1$ 因此 Q_0变到 1。　　$J_1 = K_1 = 0$ 因此 Q_1保持 1。

$J_2 = K_2 = 0$ 因此 Q_2保持 0。　　$J_3 = K_3 = 0$ 因此 Q_3保持 0。

在接下来的 7 个时钟脉冲，这个过程继续，也就指出计数器按照 BCD 码的序列计数。

9. 参见图 P-52。

11. 参见图 P-53。

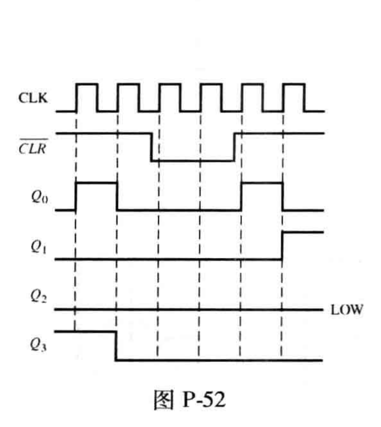


图 P-52

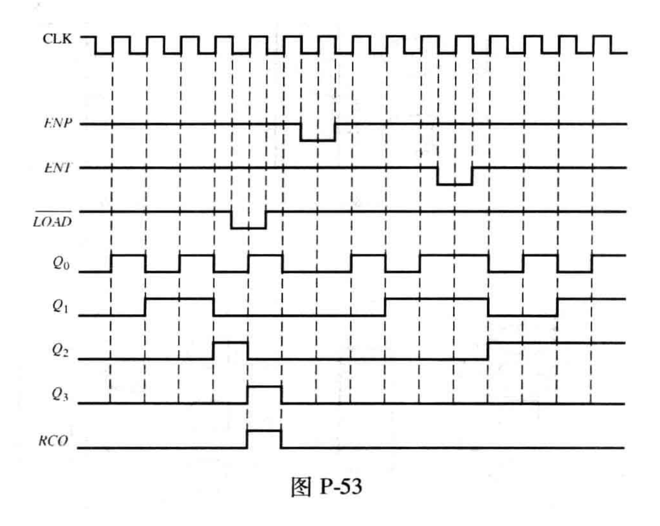


图 P-53

13. 参见图 P-54。

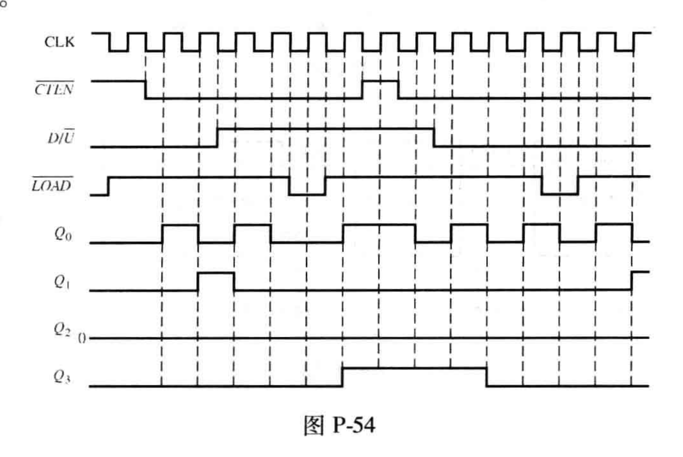


图 P-54

15. 参见图 P-55。

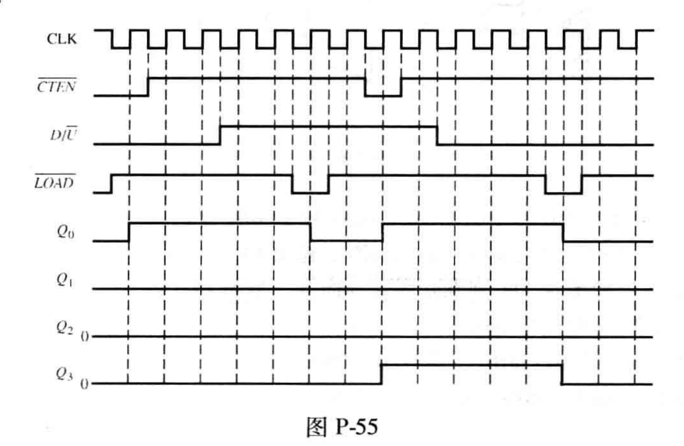


图 P-55

17. 序列为 0000，1111，1110，1101，1010，0101。计数器“自锁”在 1010 和 0101 状态，并在这两个数之间变化。
19. 参见图 P-56。
21. 参见图 P-57。
23. 参见图 P-58 的 10 000 分频电路。再增加一个除 10 计数器以产生 100 000 分频。

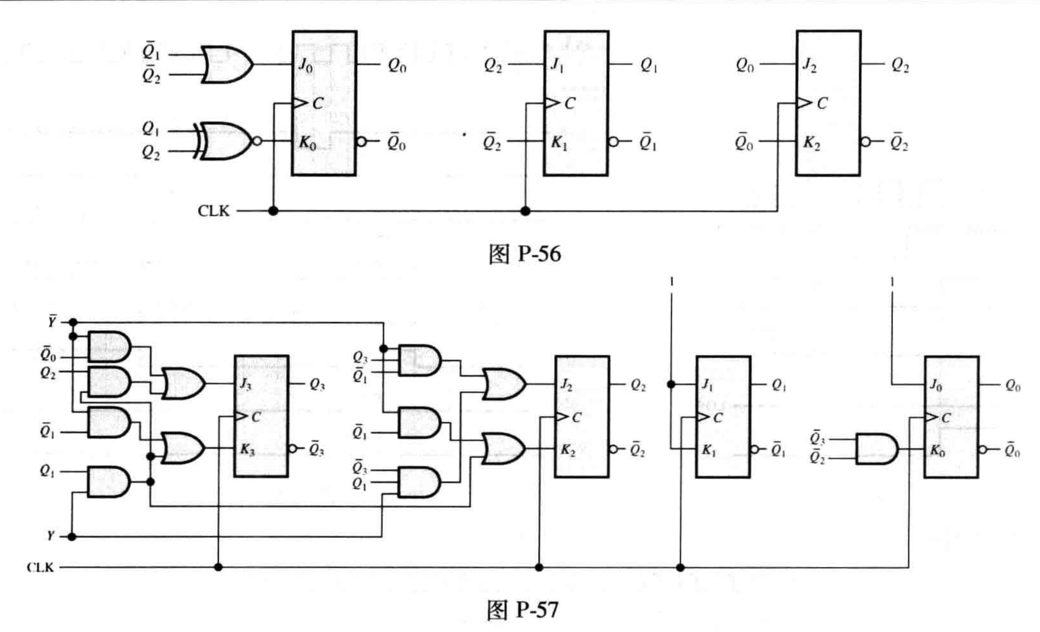


图 P-56

图 P-57

25. 参见图 P-59。

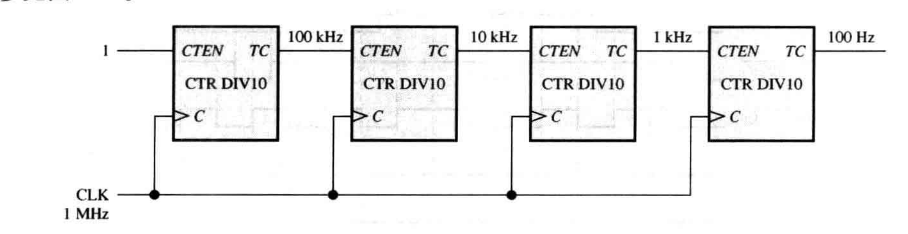


图 P-58

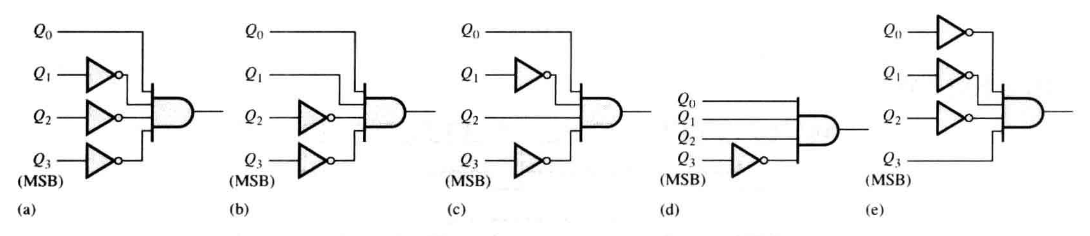


图 P-59

27. CLK2，输出 0；CLK4，输出 2，0；CLK6，输出 4；CLK8，输出 6，4，0；CLK10，输出 8；CLK12，输出 10，8；CLK14，输出 12；CLK16，输出 14，12，8。

29. 与门输出的假信号发生在 111 到 000 的转换时间。通过和计数器输出相与(选通)或使用格雷码消除假信号。

31. 小时 10：0001　小时单位：0010　分钟 10：0000
分钟单位：0001　秒 10：0000　秒单位：0010

33. 68

35. 参见图 P-60。

37. 增加 25 s 单稳的时间常数 $R_{EXT}C_{EXT}$，增加 2.4 倍。

39. 参见图 P-61。

41. 参见图 P-62。

43. 参见图 P-63。

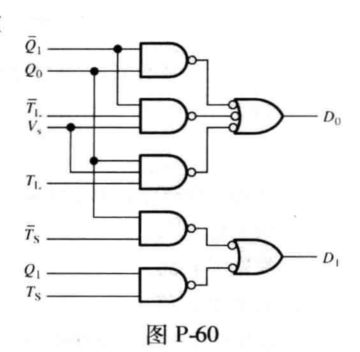


图 P-60

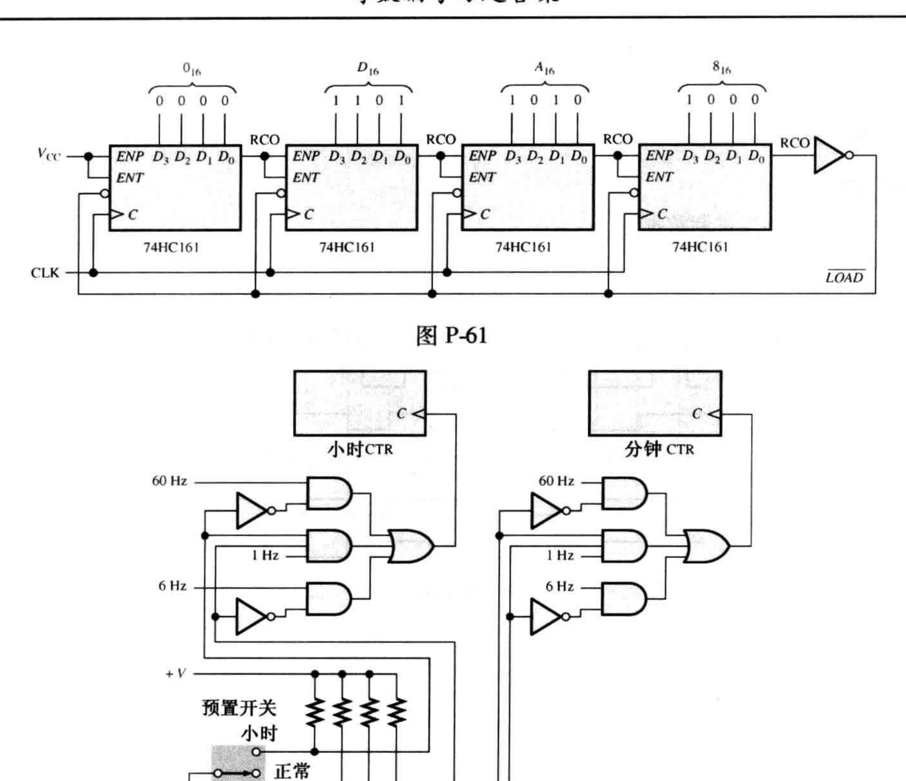


图 P-61

图 P-62

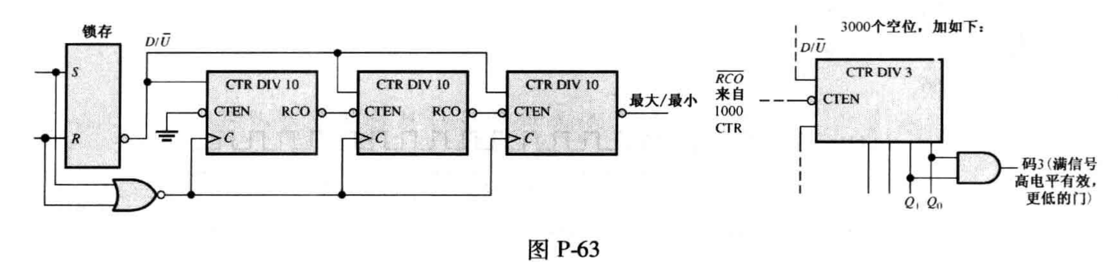


图 P-63

45. 参见图 P-64。

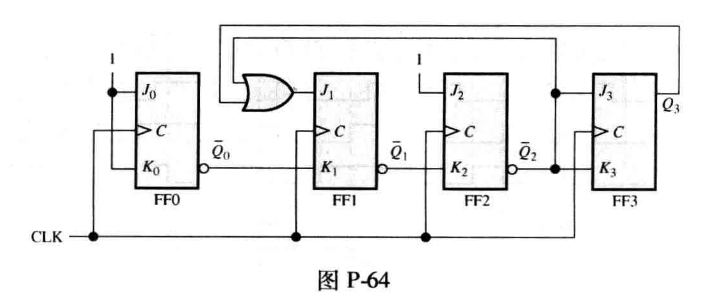


图 P-64

47. Q 输出 U_3 开路。

49. G_3 的引脚 A 开路。

51. 引脚 9 开路。

第 9 章　移位寄存器

1. 移位寄存器存储二进制数据。

3. 移位数据和存储数据。

5. 参见图 P-65。

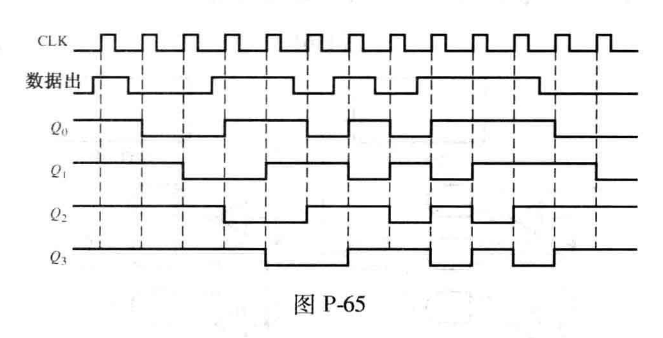


图 P-65

7. 初始时：101001111000

   CLK1：010100111100

   CLK2：001010011110

   CLK3：000101001111

   CLK4：000010100111

   CLK5：100001010011

   CLK6：110000101001

   CLK7：111000010100

   CLK8：011100001010

   CLK9：001110000101

   CLK10：000111000010

   CLK11：100011100001

   CLK12：110001110000

9. 参见图 P-66。

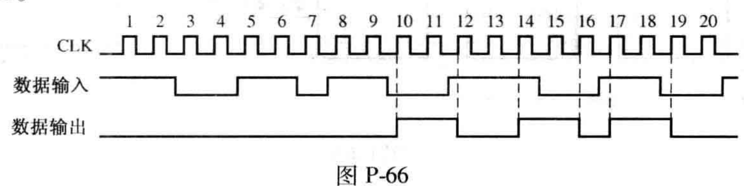


图 P-66

11. 参见图 P-67。

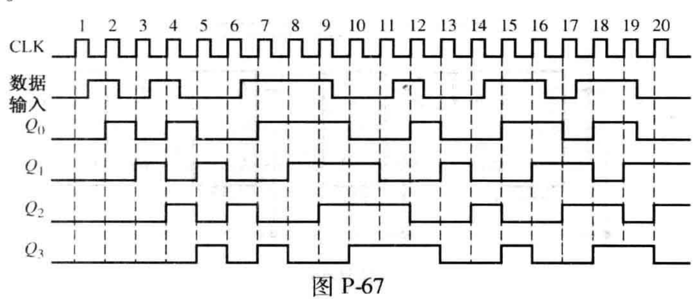


图 P-67

13. 参见图 P-68。

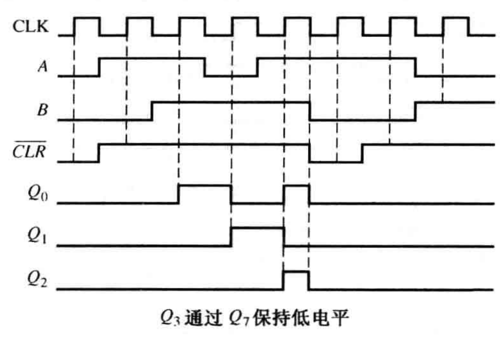


图 P-68

15. 参见图 P-69。

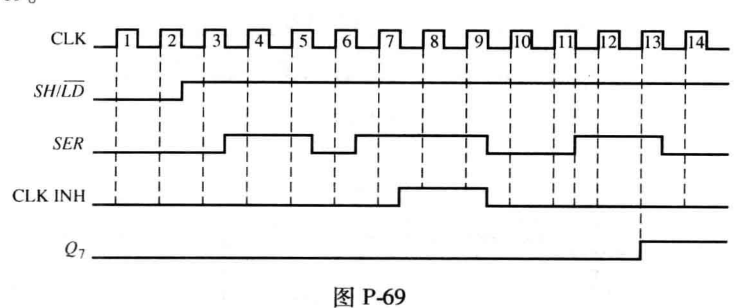


图 P-69

17. 参见图 P-70。

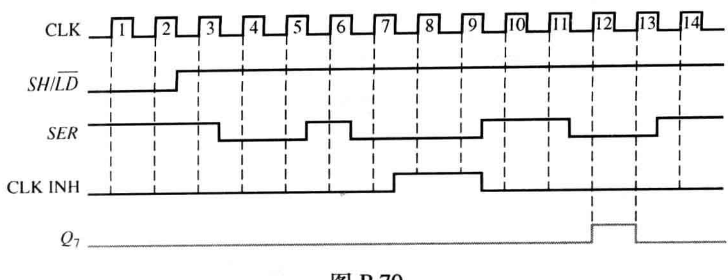


图 P-70

19. 参见图 P-71。

21. 初始时（76）：01001100

CLK1：10011000 左

CLK2：10011000 右

CLK3：00100110 右

CLK4：00010011 右

CLK5：00100110 左

CLK6：01001100 左

CLK7：00100110 右

CLK8：01001100 左

CLK9：00100110 右

CLK10：01001100 左

CLK11：10011000 左

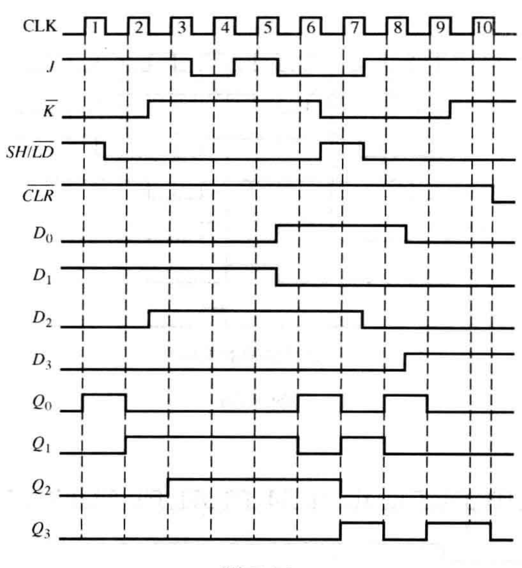


图 P-71

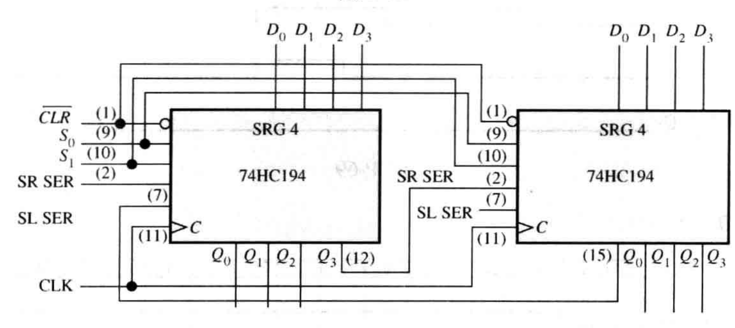


图 P-72

23. 参见图 P-72。

25. (a)3 (b) 5 (c) 7 (d) 8

27. 参见图 P-73。

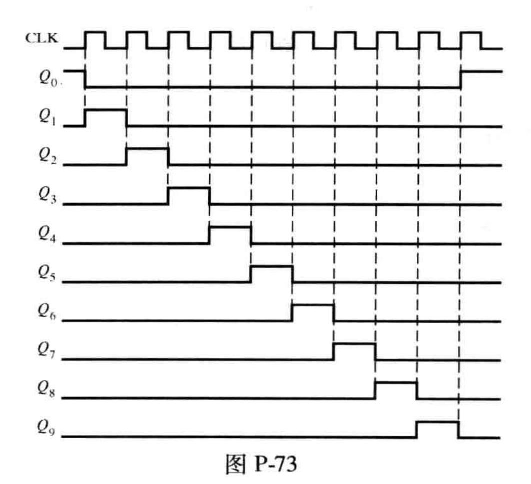


图 P-73

29. 参见图 P-74。

31. 可能产生一个不正确的码。

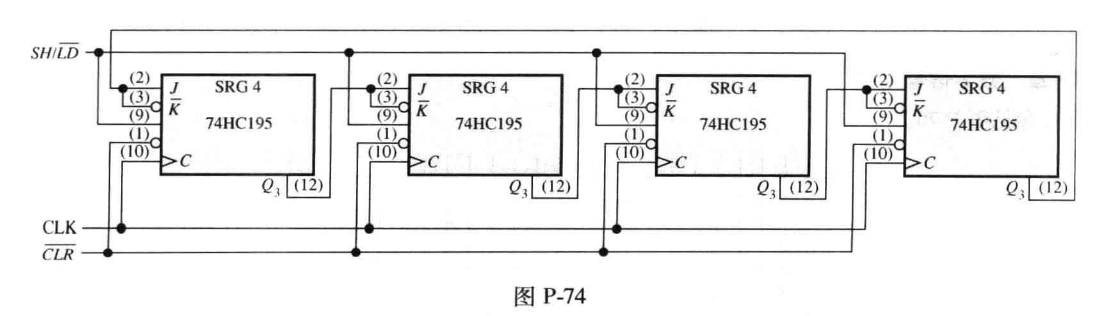


图 P-74

第 10 章　内存和外存

1. (a)ROM　(b) RAM

3. 地址总线提供了地址代码到存储器的传送，以便于以任何顺序对存储器的任何位置进行读或写的操作。数据总线提供微处理器和存储器或 I/O 之间的数据传送。

5.

	位 0	位 1	位 2	位 3
行 0	1	0	0	0
行 1	0	0	0	0
行 2	0	0	1	0
行 3	0	0	0	0

7. 512 行 × 128 的 8 位列。

9. 只要电源没有断开，SRAM 触发器中的位永远保存。DRAM 将位存储在电容器中，但必须周期性的刷新才能保留数据。

11. 参见表 P-10。

13. 参见图 P-75。

表 P-10

输入		输出			
A_1	A_0	O_3	O_2	O_1	O_0
0	0	0	1	0	1
0	1	1	0	0	1
1	0	1	1	1	0
1	1	0	0	1	0

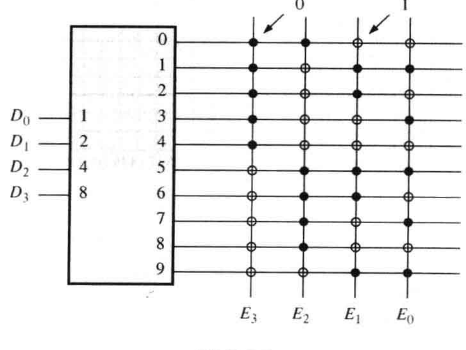


图 P-75

15. 使用 16 条地址线的 8 个 16k × 4 DRAM。其中两条地址线被译码用做存储器片选的使能输入。4 条数据线与每一个芯片相连。

17. 8 位，64k 字；4 位，256k 字。

19. 最低地址：$FC0_{16}$；最高地址：FFF_{16}。

21. 一个硬盘格式化为磁道和扇区。每个磁道分为若干个扇区，每个扇区有一个物理地址。典型的硬盘有几百到几千个磁道。

23. 磁带比磁盘有一个较长的范围时间，因为数据必须顺序访问，而不是随机访问。

25. 0010

27. 通过简单的移动 DIP 开关设置，可以输入一个新的码；不需要电源。

29. 添加一个开关到存储器(J5)，增加4与门，把或门变为5输入的，把寄存器变为5位的。

第11章 数字信号处理

1. 参见图 P-76。

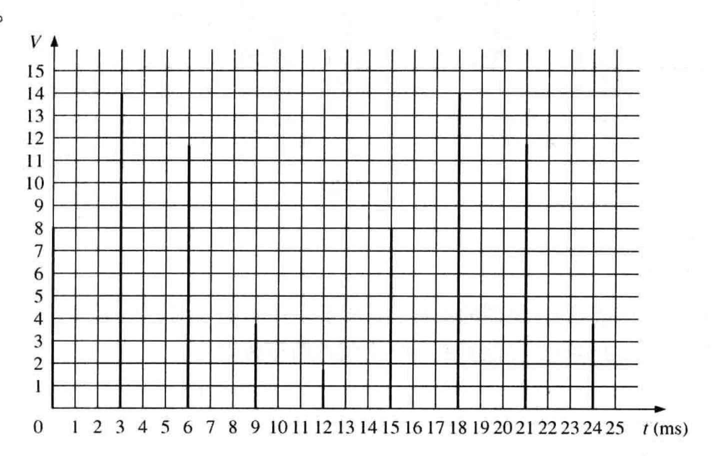
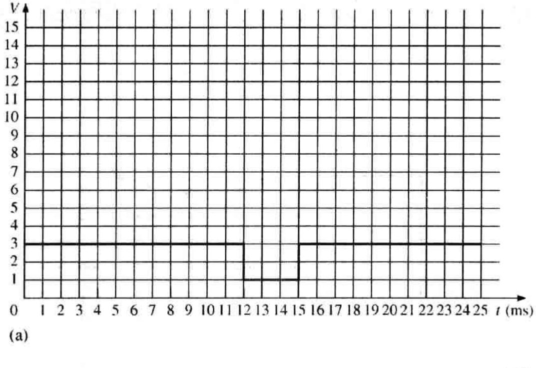


图 P-76

3. 11，11，11，11，01，11，11，11，11

5. 参见图 P-77。



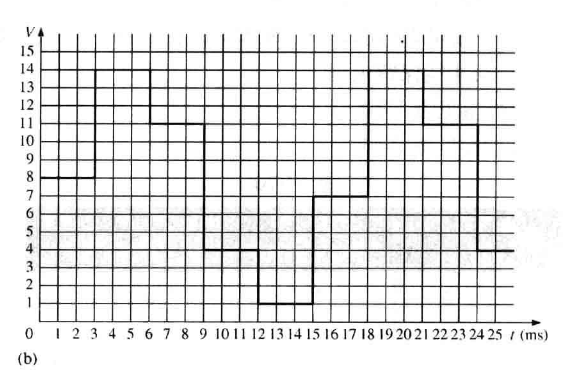


图 P-77

7. 200

9. −21.4

11. 001，010，011，101，110，111，111，111，111，110，
101，101，110，110，110，101，100，
011，010，001

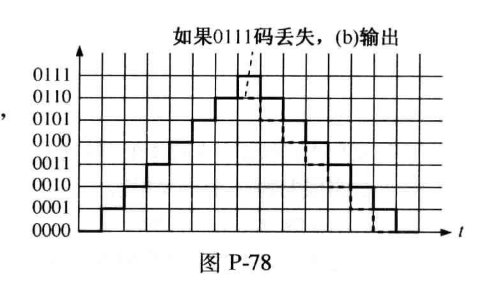


图 P-78

13. 11，11，11

15. 参见图 P-78。

17. 参见图 P-79。

19. (a) 14.3%　(b) 0.098%　(c) 0.00038%

21. 参见图 P-80。

23. 一个模-数转换器将一个模拟信号转换成为数字码。

25. 一个数-模转换器将一个数字代码转换成为相应的模拟信号。

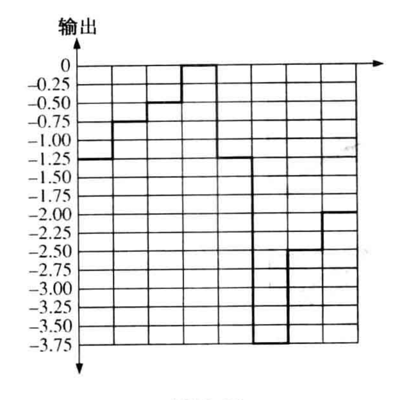


图 P-79

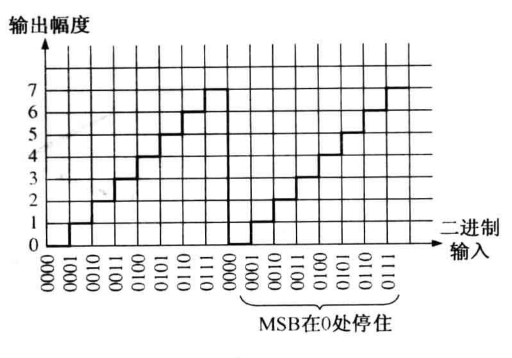


图 P-80

第 12 章　集成电路技术

1. 无；$V_{OH(min)} < V_{IH(min)}$

3. 0.15 V 处在高电平状态；0.25 V 处在低电平状态。

5. 门 C　　7. 12 ns　　9. 门 C　　11. 是，G_2

13. (a) 导通　(b) 截止　(c) 截止　(d) 导通

15. 对于一个可能的电路，参见图 P-81。

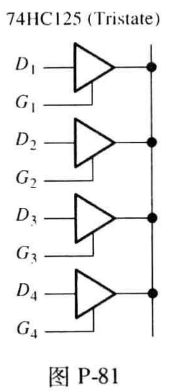


图 P-81

17. (a) 高电平　(b) 悬浮　(c) 高电平　(d) 高阻

19. (a) 低电平　(b) 低电平　(c) 低电平



21. 参见图 P-82。



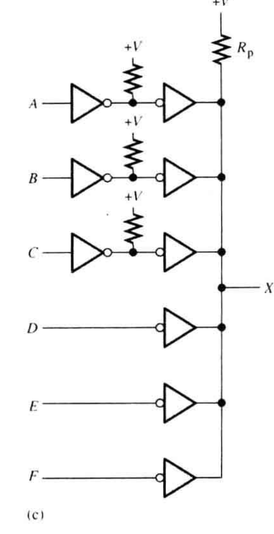




图 P-82

23. (a) R_p = 198 Ω　(b) R_p = 198 Ω　(c) R_p = 198 Ω

25. ALVC

27. (a) A, B 到 X: 9.9 ns　C, D 到 X: 6.6 ns

(b) A 到 X_1, X_2, X_3: 14 ns　B 到 X_1: 7 ns　C 到 X_2: 7 ns　D 到 X_3: 7 ns

(c) A 到 X: 11.1 ns　B 到 X: 11.1 ns　C 到 X: 7.4 ns　D 到 X: 7.4 ns

29. ECL 是 BJT 工作在非饱和状态下。